ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
Grado en Ingeniería del Software

Detección de rutinas: Internet of People. Prueba de concepto

Routine detection: Internet of People. A proof of concept

Realizado por
Alejandro Pérez Vereda
Tutorizado por
Carlos Canal Velasco
Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA
MÁLAGA, Julio 2015

Fecha defensa:
El Secretario del Tribunal

Agradecimientos

Deseo expresar mi más sincero agradecimiento al Profesor D. Carlos Canal Velasco, tutor de la presente memoria, por su estímulo, dedicación y valiosas orientaciones que me ha ofrecido en todo momento.

A mis compañeros de la Universidad de Extremadura por la colaboración y ayuda recibidas, en especial a Pablo Pérez Lozano y Juan Manuel Murillo Rodríguez.

A mi amigo Albert Munné Fuentes por la ayuda recibida con el diseño tanto de la aplicación como de la presente memoria.

Por último, un agradecimiento especial a mi familia por el apoyo durante toda la elaboración del trabajo.

Resumen: El principal objetivo de Internet of Things (IoT) es integrar las tecnologías informáticas en el quehacer cotidiano de las personas, facilitando su interacción con un entorno de dispositivos interconectados, pero el estado actual del arte hace que dicha interacción esté aún lejos de resultar trivial, precisando de continua intervención del usuario. Como alternativa a esta situación, iniciativas emergentes como la de Internet of People (IoP) pretenden integrar de forma más efectiva el IoT en la vida de las personas. En línea con este propósito, el modelo People as a Service (PeaaS) facilita estas tareas por medio del uso del teléfono móvil como interfaz del usuario con el IoT y haciendo uso del contexto del usuario del mismo. PeaaS permite elaborar un perfil sociológico del usuario, que puede ser explotado por el mismo y servido a terceros de forma segura y controlada. En este trabajo presentamos una aplicación móvil para la supervisión de personas afectadas de alzhéimer mediante el aprendizaje y monitorización de sus rutinas como prueba de concepto del modelo PeaaS, teniendo como resultado una funcionalidad que va mucho más allá de la ofrecida por otros productos similares en este campo, y una tecnología que es base para infinidad de aplicaciones que provoquen el avance hacia IoP.

Palabras claves: Internet of People, People as a Service, Detección de rutinas, Alzhéimer, Inferencia de datos.

Abstract: The Internet of Things (IoT) main point is to integrate computer technology in people's ordinary life, easing their interaction with an environment of interconnected devices, but the actual state of the art makes this interaction be far yet from being trivial, requiring a continuous user's intervention. As an alternative to this situation, emergent alternatives as the Internet of People (IoP) aim to integrate in an effective way the IoT in people life. Following this purpose, the People as a Service (PeaaS) model pretends to simplify these tasks by using the mobile phone as an user's interface with IoT and taking advantage from the mobile's user context. PeaaS enables elaborating a sociologic profile for the user, which he/she can use and offer to third parties in a safe and controlled way. In this work, we present a mobile application for the supervision of people affected with Alzheimer by learning and monitoring their routines as a proof of concept of PeaaS model, obtaining as a result a functionality that goes further from the ones which similar products today offer, and a base technology for a wide variety of mobile applications that will cause the evolution to IoP.

Keywords: Internet of People, People as a Service, Routine detection, Alzheimer, Data inference.

Índice

1. Introducción

A día de hoy, el móvil y sus aplicaciones sociales (email, mensajería, redes sociales…) se han convertido en un gran pilar de nuestra vida. Por ellas expresamos todo tipo de cosas, desde como nos sentimos o nuestras posturas éticas o sociales, hasta donde y con quien hemos estado o que hemos comido [1]. Se han convertido en nuestra principal herramienta de expresión, interacción y de obtención de información.

Un concepto del que se habla mucho en la actualidad es el Internet of Things (IoT), cuyo principal objetivo es integrar las tecnologías informáticas en el quehacer cotidiano de las personas [2, 3], haciéndoles la vida más sencilla, de forma que, por ejemplo, pueda programarse que se encienda de forma remota la cafetera utilizando el móvil para tener el café hecho al levantarnos o, incluso planificar esta tarea de acuerdo con nuestros hábitos diarios.

Sin embargo, la forma en que las tecnologías relacionadas con este concepto se aprovechan en la actualidad es notablemente susceptible de mejora. Un aspecto importante a tener en cuenta para el futuro progreso de estas tecnologías es la existencia aún de un salto entre la red donde la información es tratada e intercambiada y la realidad de la vida física y su contexto [4].

En general, el usuario necesita configurar diversos parámetros del sistema en cuestión de forma manual, parámetros que debe reconfigurar cuando el contexto cambia, de forma que lo que inicialmente se piensa como una comodidad, se vuelve algo tedioso y con una integración mínima con nuestra vida.

El resultado es que, lejos de hacer que la tecnología trabaje para las personas, esta obliga a las personas a estar pendientes de introducir nuevas órdenes o modificar la planificación siempre que se produzca un cambio inesperado en su día a día. En un escenario más adecuado, la tecnología debería tener en cuenta el contexto de las personas a las que debe servir, aprendiendo de dicho contexto y realizando acciones de forma proactiva de acuerdo con su situación y expectativas en cada momento. Así cuando la persona va a levantarse más tarde de lo habitual, le gustaría que el café empezara a hacerse en el momento justo para tomárselo caliente al levantarse, o si va a alguna parte en busca de ocio, pueda contactar con más gente interesada o saber cual es el mejor lugar para ir en ese momento.

Un escenario más adecuado, que resolviera estas deficiencias, podría ser el planteado por el modelo de People as a Service (PeaaS) [5], un modelo social que pone su énfasis en el usuario como proveedor de servicios y al que permite controlar la información que fluye de su dispositivo, dando especial importancia a aspectos de privacidad y de

seguridad. Así, nuestro móvil, que tiene toda esta información acerca de nosotros podrá aprovecharla llegando a ser capaz de inferir nuestras rutinas, comportamiento, [6,7] y nuestro contexto para realizar un perfil sociológico y poder proporcionarlo a terceros de forma segura y controlada o detectar nuestras rutinas de comportamiento y explotar esta información para hacernos más llevadero nuestro propósito diario, convirtiéndose el teléfono en nuestra interfaz con el mundo que nos rodea, de forma automática y especializada en cada uno de nosotros.

A continuación desarrollaremos esta idea según la siguiente estructura, en esta primera sección, se exponen las motivaciones y los objetivos de este trabajo y su campo de aplicación. Además se habla de las tecnologías y materiales usados durante la elaboración de este. La segunda sección de esta memoria, presenta la idea de la prueba de concepto elegida y desarrolla los principales aspectos técnicos de la misma, incluyendo entre otros la descripción del caso de estudio, la arquitectura de la aplicación, la estructura de datos utilizada, el procedimiento de análisis de detección de rutinas y el estado actual del producto. La tercera sección relata el procedimiento de desarrollo que se ha llevado a cabo. Por último, en las dos secciones que nos restan, se hace un repaso de los principales trabajos relacionados, y se recogen las conclusiones de este trabajo.

1.1 Motivación

En la actualidad, con la aparición en el mercado de diversos dispositivos *wearable*, el sector de IoT está experimentando un gran crecimiento y difuminando las fronteras que había en cuanto a lo que es posible hacer con estas tecnologías. La mayoría de estos dispositivos salen al mercado en una línea claramente dirigida hacia sanidad y hábitos saludables. Esta es una de las principales razones que nos ha llevado a elegir este sector como la prueba de concepto más adecuada. Además, este es uno de los ámbitos donde este tipo de tecnología podría ser muy útil y es fácilmente implantable.

Existen en la actualidad diversas aplicaciones móviles para la ayuda de personas con diferentes enfermedades, entre ellas, una de las que más atención se le esta prestando y en cuyo ámbito, PeaaS podría ser de gran ayuda, es el alzhéimer. De las aplicaciones más útiles y avanzadas que actualmente existen para facilitar el día a día de sus afectados podemos destacar Cerqana[1] y Tweri[2]. Sin embargo, ninguna de ellas considera el aprendizaje automático de rutinas y patrones de desplazamiento. Esto sin

[1] Cerqana. http://www.cerqana.com/
[2] Tweri. http://www.tweri.com/inicio.aspx

duda, se convierte en una gran ventaja que pretendemos que nuestra aplicación aproveche con respecto a las ya existentes, ya que minimiza la necesidad de configuración de distintos parámetros que pueden ser complejos. De esta manera, se facilita el uso de la aplicación tanto a los enfermos, que suelen tener edad avanzada y no tienen interiorizado el uso del teléfono, que quizás solo les bastaría con llevar una pulsera o reloj con GPS, como a sus cuidadores, que simplemente recibirán notificaciones cuando sea oportuno. Así estaremos usando como base el modelo de PeaaS propuesto, convirtiéndose en un cambio radical en el esquema habitual usado por la tecnología para su desarrollo. En una implementación de este tipo, nuestra propuesta debe seguir el modelo de la pirámide DIKW para la inferencia y aprendizaje a partir de la información del usuario y su contexto; que distingue entre los niveles de datos, información, conocimiento y sabiduría [8]. Por ejemplo, partiendo de la recolección y agregación de datos de geolocalización temporal del teléfono móvil, se obtiene información sobre lugares y trayectorias habituales del usuario. A partir de esta información ya será posible inferir conocimiento sobre sus rutinas de desplazamiento y frecuencia de las mismas, para llegar finalmente a adquirir la sabiduría sobre el usuario, que es necesaria para poder predecir comportamientos y detectar desviaciones en los mismos, tomando decisiones de emisión de alertas en caso necesario.

1.2 Objetivos

El principal objetivo de este trabajo es colaborar en el desarrollo de una plataforma software que mejore cómo las personas se integran en IoT, para empezar a hacer uso de la información contextual y el perfil de los usuarios alojados en sus teléfonos móviles. Esto abre un camino a nuevos escenarios de IoT que permitan la evolución a Internet of People [9] (IoP). Esta línea de investigación parte de trabajos anteriores sobre los modelos y plataformas Social Devices [10, 11] y PeaaS. Social Devices es una plataforma desarrollada inicialmente por Nokia y la Universidad de Tampere que tiene por objeto enriquecer, incrementar y facilitar las interacciones entre personas geográficamente próximas y entre ellas e IoT. Por otro lado, PeaaS, es un modelo y plataforma de computación móvil que permite la gestión de los perfiles sociológicos de los usuarios, que son inferidos y almacenados en el propio dispositivo móvil y proporcionados a terceros como un servicio en la Nube, de manera segura y controlada por su propietario. La combinación de ambos modelos permite convertir el teléfono móvil en la interfaz natural entre las personas e IoT, de forma que este almacena toda la información contextual necesaria, detecta desviaciones de las rutinas

habituales e interactúa con IoT en consecuencia, minimizando la necesidad de interacciones del usuario.

Con la intención de validar este modelo, y como prueba de concepto del mismo, este trabajo se centra en el desarrollo de una aplicación Android para el seguimiento de personas con capacidades intelectuales mermadas, pero con cierto grado de autonomía, como pueden ser las personas con alzhéimer. El objetivo es que el teléfono móvil aprenda los patrones habituales del usuario (horarios, desplazamientos, etc.), obteniendo un perfil de las rutinas de desplazamiento del mismo, y de forma que pueda supervisar dichas actividades y realizar acciones (por ejemplo, alertas al paciente o a sus familiares) en caso de un cambio inesperado en dichos patrones. La propuesta encaja con el interés actual en el desarrollo de productos software relacionados con la salud de los usuarios. Es tal el crecimiento de este sector que incluso Google ofrece una API[3] para la creación de aplicaciones destinadas a fomentar hábitos saludables de una forma más fácil, rápida y eficiente.

Aunque inicialmente el escenario se desarrollará en una plataforma basada en teléfonos móviles, la proyección inmediata sería su uso en otros tipos de dispositivos como los wearables, los cuales están teniendo una gran aceptación actualmente y serían mucho más cómodos y personales para sus usuarios.

1.3 Tecnologías utilizadas

En la elaboración del presente trabajo se ha precisado de equipos informáticos tanto para la elaboración de la memoria como para la elaboración de la aplicación prueba de concepto. Para la aplicación se ha optado por desarrollarlo para la plataforma Android, por lo que se ha usado software necesario para el desarrollo, en nuestro caso hemos usado la herramienta propia de este sistema, que es Android Studio[4]. Android Studio es el IDE oficial para desarrollo Android basado en IntelliJ IDEA. Este IDE, entre otras cosas, trae un soporte para Google Cloud Platform, lo cual permite una fácil integración con App Engine y con Google Cloud Messaging; este último servicio permite a los desarrolladores enviar mensajes en diferentes plataformas, como Android, iOS y Chrome y está integrado en nuestro proyecto mediante la plataforma Nimbees[5].

[3] Google Fit. http://developers.google.com/fit/
[4] Android Studio. https://developer.android.com/sdk/index.html
[5] Nimbees. http://www.nimbees.com/

Al tratarse en definitiva de dos aplicaciones, para que el cuidador pueda monitorizar al usuario, también se ha precisado el uso de la plataforma Nimbees. Esta es una plataforma de notificaciones push con funcionalidades adicionales a las habituales, como un potente motor de segmentación basado en una tecnología de gestión de perfiles de usuarios en el dispositivo, que nos permite realizar filtros y segmentaciones de usuario. Además cuenta con una opción de cuenta gratuita que nos permite una monitorización de todos los usuarios y mensajes enviados de forma cómoda vía web. La API también es bastante sencilla de implantar y usar en la aplicación.

Figura 1. Tecnologías utilizadas.

Durante la implementación hemos hecho uso de las librerías propias SDKs además de otras, principalmente las de Google Maps.

En la figura 1 puede observarse un compendio de las tecnologías más importantes que han sido utilizadas en la elaboración de este trabajo.

Durante la fase de desarrollo se investigó el uso de otro tipo de APIs como es la descrita al final del apartado anterior, Google Fit, para el manejo de datos; y algunos framework de inferencia de datos, pero finalmente se descartaron debido a que los análisis a realizar no requerían una plataforma mayor y se desarrolló un algoritmo de aprendizaje y análisis propio que es suficiente para la tarea.

2. Una aplicación para enfermos de alzhéimer

El modelo PeeaS es una plataforma de computación móvil basada en la obtención de información del usuario para su posterior tratamiento y así conseguir un perfil sociológico del mismo, que será almacenado en su propio teléfono móvil, proporcionándole una mayor intimidad, seguridad e integridad.

Estos perfiles serán gestionados en el propio dispositivo y ofrecidos a terceros, que pueden ser otros usuarios, como un servicio de la Nube de manera segura y controlada por su propietario, permitiendo crear una red de información verídica y en tiempo real.

El escenario elegido como prueba de concepto de PeaaS consiste en el desarrollo de CareMe, una aplicación Android para ayudar a personas con alzhéimer en las fases iniciales de esta enfermedad. Con ella se quiere contribuir a mejorar su calidad de vida y la de sus cuidadores.

El síndrome de Alzheimer, es una enfermedad neurodegenerativa primaria que produce un deterioro cognitivo y trastornos conductuales. La persona con alzhéimer experimenta cambios microscópicos en el tejido de ciertas partes de su cerebro y una pérdida, progresiva, pero constante, de una sustancia química, vital para el funcionamiento cerebral, llamada acetilcolina. Esta sustancia permite que las células nerviosas se comuniquen entre ellas y está implicada en actividades mentales vinculadas al aprendizaje, memoria y pensamiento.

La enfermedad de Alzheimer es la forma más común de demencia, es incurable y terminal, y aparece con mayor frecuencia en personas mayores de 65 años de edad.

Esta afecta al enfermo con síntomas como la pérdida de memoria, desorientación o la pérdida de otras capacidades mentales que pueden resultar muy peligrosas para su seguridad y la realización de su vida diaria. Estos síntomas pueden manifestarse con mayor fuerza en ciertos momentos, afectando a la persona de inmediato. Por ejemplo, puede darse el caso en que el enfermo ha salido de casa a realizar alguna actividad, comprar el pan por ejemplo, y durante ella olvida el propósito de lo que estaba haciendo o cómo volver a casa.

Figura 2. Cifra de personas con alzhéimer en el mundo.

Según se puede ver en la figura 2, existen alrededor de 47,5 millones de personas afectadas por alzhéimer en el mundo, y se estima que esta cifra alcanzará los 80 millones dentro de 20 años. Esta enfermedad no solo afecta a los pacientes, sino que también condiciona la vida de sus cuidadores, ya que les obliga a estar constantemente pendientes de ellos y preocupados por su situación.

Con objeto de contribuir a la mejora de esta situación, nos planteamos el desarrollo de una aplicación para dispositivos móviles basada en PeaaS y capaz de aprender las rutinas de desplazamiento del usuario**.** La motivación para ello fue doble. Por un lado la penetración cada vez mayor de los dispositivos móviles en la sociedad incluso entre la población de edad más avanzada; además con la creación de nuevos dispositivos que cada vez son más personales y se integran de forma más discreta y cómoda en nuestra vida, como los relojes, pulseras, gafas... Por otro, las ventajas que proporciona PeaaS para potenciar el uso del contexto de los usuarios de estos dispositivos móviles y mejorar la integración de dichos usuarios con su entorno.

La aplicación diseñada cuenta con dos componentes funcionales distintos, pero comunicados entre sí, en función del tipo de usuario, puede ser un paciente enfermo de alzhéimer o su cuidador:

- Dispositivo Paciente: Es el componente principal y en el que se centra la actividad de PeaaS. Se trata de una aplicación Android que reside en el dispositivo del paciente. Valiéndose de la información facilitada por el GPS del dispositivo, la aplicación registra los desplazamientos del usuario para posteriormente analizarlos y ser capaz de aprender y detectar las rutinas de desplazamiento diarias del usuario, llegando así incluso a predecir a donde va a dirigirse el usuario y en que momento del día lo hará. La utilidad de la aplicación viene al ir comprobando continuamente si los desplazamientos que va realizando son acordes a dichas rutinas, y en caso contrario, emitir avisos tanto al paciente como a su cuidador según diferentes niveles de alarma que pueden ser configurados. CareMe utiliza, analiza y pone al servicio de otros su contexto. En este caso los otros serían los cuidadores, que de forma segura podrán acceder a la información de los desplazamientos del paciente.

careme
cuida de tu familia

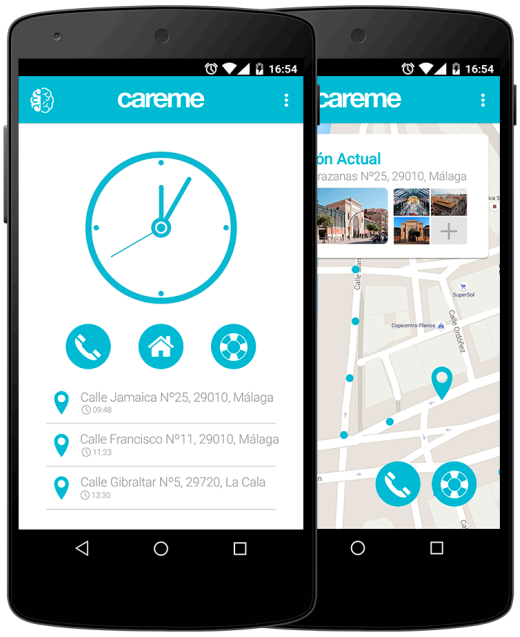

Figura 3. Aplicación CareMe.

- Dispositivo Cuidador: De nuevo es una aplicación Android, que en este caso reside en el dispositivo del cuidador. A través de ella el cuidador puede realizar un seguimiento de la posición del paciente y configurar de forma remota la aplicación de este, indicando los niveles de alarma deseados, las actuaciones a

llevar a cabo en la detección de cada nivel, o introduciendo nuevos compromisos o citas puntuales que el paciente pueda tener, como una cita con el médico. La aplicación además se encarga de recibir y procesar las alarmas emitidas por el otro componente dándoles el tratamiento adecuado y mostrándolas al usuario, permitiéndole realizar una serie de acciones en reacción a esta como un seguimiento en tiempo real, llamar al paciente, o llamar a urgencias de forma rápida.

En la figura 3 se muestra una captura de ambas aplicaciones con el diseño final de sus interfaces gráficas.

2.1 Actores

Habrá dos actores además del sistema que tendrán acceso a este.

El usuario principal, que será el paciente de alzhéimer. Es sobre quien el sistema trabaja y obtiene las rutinas, pero recordemos que la idea es tratar de reducir al máximo la interacción de este con el sistema, tratando que la introducción de datos sea mínima o nula, es el sistema automáticamente quien deberá aprender del paciente sin que este se dé cuenta si quiera. Sin embargo, si se le permitirá en cualquier caso hacer algunas consultas en el móvil, o realizar algunas acciones de emergencia como alertar al cuidador, o iniciar una navegación a casa.

El otro actor es el responsable del usuario, que nosotros hemos denominado cuidador (caregiver, en inglés) que tendrá todo el control sobre la supervisión del anterior, además de poder realizar cambios en el sistema, llegando incluso a poder introducir citas concretas a las que el usuario deberá acudir para que se consideren como una rutina más, pero pueden ser sin ningún tipo de periodicidad.

2.2 Requisitos

A continuación se citan los requisitos funcionales y no funcionales a ser tenidos en cuenta en la producción de la aplicación con el fin de intentar definir de forma más concreta el sistema con los requerimientos mínimos que este debería cumplir.

Requisitos Funcionales Mínimos

1. El usuario podrá consultar sus rutinas. Al abrir la aplicación se le mostrará una lista de los lugares a los que asistirá ese día y a la hora a la que tendrá que salir para llegar a ellos.

2. El sistema creará notificaciones para cada situación según su riesgo. Si el paciente se pierde o entra en situación de riesgo, el sistema se encargará de detectarlo y emitir una alerta al usuario con opción de iniciar una navegación a casa.
3. El sistema avisará a los contactos de emergencia en situaciones de riesgo. Si el paciente se pierde o entra en situación de riesgo, el sistema se encargará de detectarlo y enviar la correspondiente alerta al cuidador.
4. El sistema calculará constantemente las rutinas del usuario. El sistema será capaz de aprender, sin necesidad de interacción del usuario, sus rutinas habituales de desplazamiento diarias y actualizarlas de forma automática frente a cualquier cambio en ellas.
5. El sistema gestionará las rutinas del usuario.
    5.1. El sistema consultará rutinas
    5.2. El sistema creará rutinas.
    5.3. El sistema actualizará rutinas.
    5.4. El sistema eliminará rutinas.
6. El sistema detectará desvíos en la rutina del usuario. El sistema monitorizará todos los movimientos del paciente para contrastarlos con sus rutinas y detectar en su caso el desvío del paciente si se sale de ellas.
7. El sistema almacenará los movimientos del usuario. El sistema mantendrá un histórico de los desplazamientos realizados por el paciente para su análisis y recuperación en caso de fallo.
8. El sistema eliminará datos antiguos. El sistema deberá borrar datos del histórico que ya sean bastante antiguos y hayan sido tenidos en cuenta en el análisis con la intención de optimizar el uso de la memoria del teléfono.
9. El responsable podrá recibir notificaciones del sistema. El dispositivo del cuidador recibirá notificaciones de alertas del paciente si este se desvía de una rutina o está en peligro porque se ha perdido.

Requisitos Funcionales Opcionales

1. El usuario podrá actualizar sus notificaciones ajustando su intensidad. El usuario podrá establecer si quiere que el dispositivo quede en silencio, solo vibre, o vibre y suene, según el tipo de alarma.

2. El responsable podrá gestionar las rutinas, con la frecuencia que tendrán y el lugar y la hora en la que tendrán lugar. El cuidador podrá introducir modificar o eliminar cualquier rutina, dándole una frecuencia, o sin ella, siendo una cita puntual del paciente.
    2.1. El responsable podrá consultar rutinas.
    2.2. El responsable podrá crear rutinas.
    2.3. El responsable podrá actualizar rutinas.
    2.4. El responsable podrá eliminar rutinas.
3. El responsable podrá gestionar la lista de contactos de emergencia del usuario. El responsable tendrá el control sobre la lista de contactos de emergencia con los que el paciente podrá contactar en caso de algún problema o de que se haya perdido.
    3.1. El responsable podrá crear un nuevo contacto.
    3.2. El responsable podrá actualizar un contacto.
    3.3. El responsable podrá eliminar un contacto.
    3.4. El responsable podrá consultar los contactos.

Requisitos No Funcionales

1. El sistema debe estar desarrollado en plataforma Android. Es la plataforma que se ha elegido para el desarrollo en un principio, debido a que se trata de la plataforma más asequible para los usuarios, con el objetivo de que cualquiera pueda tener acceso a la aplicación sin realizar una gran inversión monetaria.
2. El sistema debe cumplir con la LOPD. Es obligatorio cumplir con la Ley Orgánica de Protección de Datos ya que trabajamos con datos sensibles de los usuarios.
3. El sistema se adaptará a los recursos disponibles. El sistema debe ser lo más eficiente posible optimizando la utilización de los recursos del dispositivo móvil, que como sabemos, son escasos.

Aunque todos estos son requisitos para una aplicación final comercial, hay algunos de ellos que se han considerado de menor relevancia para este estudio, y aún no están integrados en la fase actual de la aplicación, son los denominados opcionales. Sin embargo deberán ser tenidos en cuenta para las versiones futuras de la misma y disponer de una App de funcionalidad completa.

2.3 Arquitectura

El sistema se ha desarrollado con el mayor nivel de autonomía posible. Este es capaz de tomar decisiones y adaptarse a los recursos y situaciones que se den en el dispositivo. Para cumplir con esto y con los requisitos anteriormente descritos se ha desarrollado el sistema de forma que es capaz de realizar varias tareas independientes y de forma simultánea.

Si examinamos detenidamente los requisitos, podemos ver que la aplicación en sí, debe realizar cuatro tareas fundamentales que engloban todos los requisitos. Estas cuatro tareas que distinguimos son la de monitorización, que es la que comprende toda la detección de desvíos del usuario y la vigilancia del mismo; la de acumulación, que consiste en ir almacenando el histórico de movimientos que luego se analizará; la de análisis, que es la que podemos llamar tarea de aprendizaje, usa los datos recogidos por la tarea de acumulación para llegar a obtener las rutinas del usuario; y por último, la de actuación, que es la encargada de mostrar notificaciones y enviar alertas a los contactos de emergencia si se diera el caso.

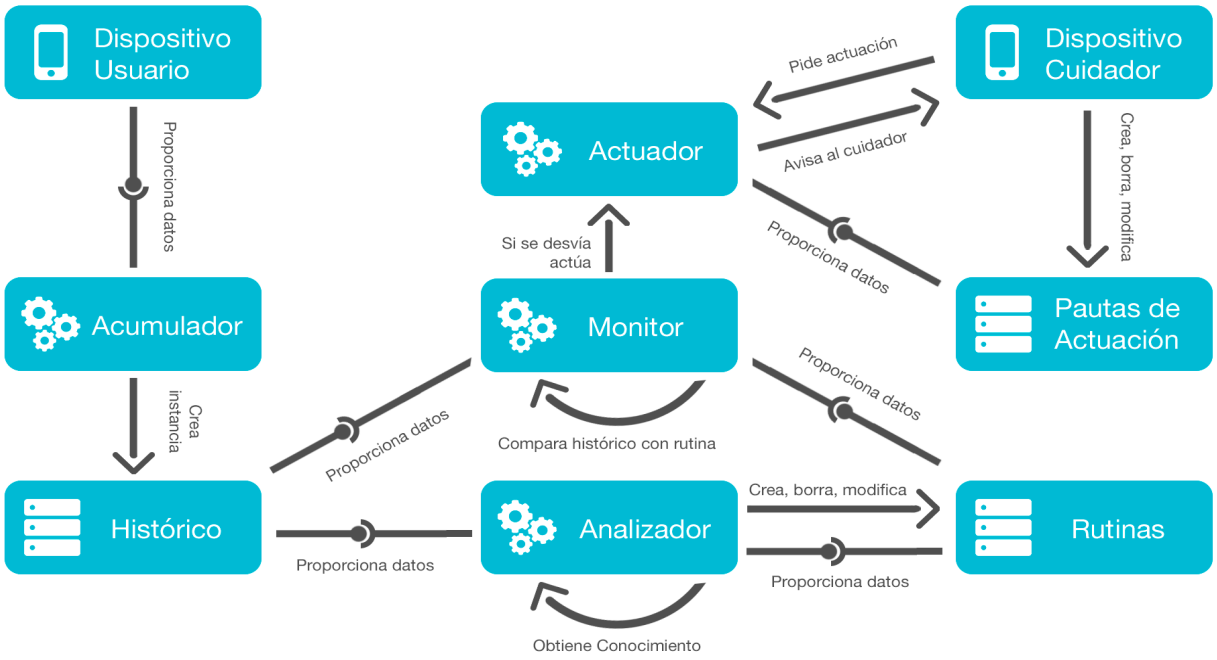

Figura 4. Arquitectura de la aplicación.

Una vez definidas las distintas tareas, las únicas que realmente necesitan ser simultáneas son la de monitorización y la de acumulación, ya que necesitan estar en funcionamiento constante durante todo desplazamiento. Es por esto, por lo que se decidió usar una arquitectura basada en tres componentes, con diferentes

características, pero comunicados entre sí de la forma en que se muestra en la figura 4. Estos componentes son:

Monitorización/Acumulación

Este es el componente que como hemos dicho consta de dos tareas que se van realizando de forma simultanea. Estas tareas son la de acumulación de histórico y la de monitorización de los desplazamientos del usuario.

Al ser un componente que necesita estar en ejecución de forma constante, se decidió implementarlo en forma de un servicio Android, que permanece activo aunque se cierre la ventana de la aplicación, así conseguimos que las dos tareas que realizan no sean interrumpidas y realice correctamente su trabajo.

La forma en que se decidió implementar este componente es con un Listener, en este caso de Localización. Este Listener nos permite configurarlo de varias maneras para poder ajustarlo a nuestros requerimientos. El que nos atañe, como va a estar activo todo el tiempo, nos interesa que el GPS que vaya a utilizar consuma lo mínimo posible, pero siempre guardando un cierto compromiso con la exactitud de la posición del usuario. Una vez configurado así, el Listener usará la mejor opción disponible atendiendo a esos requisitos. El último aspecto a configurar que queda es si necesitamos una muestra cada cierto tiempo, o cada vez que se desplace cierta distancia.

Para esta tarea se escogió la opción de coger una muestra cada cierta distancia, por el ahorro de batería y memoria, la otra opción sería antagónica. De esta forma, mientras el usuario permanezca quieto en casa, dentro de la distancia especificada al Listener, conseguimos que este componente esté en reposo, ni monitoriza ni acumula histórico, ya que sería inútil. Además esta forma de acumular histórico, es mucho más eficiente ya que solo recogemos muestras importantes, y esto es de bastante ayuda a la hora de analizarlas, haciendo dicho proceso bastante más simple.

Bien, pues ya definida la estructura de este componente y su forma de actuar, podemos explicar, como realmente trabaja.

Para empezar, la acumulación es bastante fácil, cada vez que el usuario recorre la distancia *x* expresada anteriormente, se produce una llamada al Listener, y es aquí donde aprovechamos para esta primera tarea, recogemos una muestra de latitud y longitud y la almacenamos en cuanto a una fecha y una hora en la base de datos.

Justo después de esta operación, aprovechando la misma llamada, llevamos a cabo el proceso de monitorización.

El proceso de monitorización esta pensado para ser lo más eficiente y flexible posible, refiriéndonos con flexible, a la flexibilidad de movimiento del paciente a la hora de realizar sus desplazamientos por distintos caminos, por ejemplo. Esto se consigue de la siguiente forma.

Como tenemos que aprovechar la llamada del Listener para actuar, el algoritmo se diseñó de forma que no cada vez que el usuario avanza unos pocos metros se haga todo el proceso de monitorización, lo cual sería muy costoso. Lo que se implementó fue una especie de contador, de forma que pudiéramos hacer la monitorización en función no solo de una distancia, si no de un tiempo en recorrerla también. Explicándolo mejor, el proceso de monitorización solo actúa, en el caso concreto de esta fase de la app, si en cinco minutos, el usuario ha recorrido unos doscientos metros o más; y esto lo conseguimos incrementando un contador cada vez que hay una llamada al Listener, si en una llamada han pasado mas de cinco minutos desde la última monitorización, se comprueba que al menos haya habido un número *x* de llamadas al Listener antes recogidas en el contador. Así conseguimos el mínimo consumo de batería con un rango de detección de desvíos bastante aceptable.

Ahora bien, para pasar a la explicación de cómo se detecta un desvío, hay varias cuestiones que resolver con antelación.

La primera cuestión puede estar bastante relacionada con la libertad del usuario, la capacidad de adaptación del monitor y con los niveles de riesgo que vamos a considerar. En esta implementación, se ha decidido que la monitorización no solo tenga en cuenta la rutina que tiene que estar haciendo el usuario a cierta hora hoy, sino que consideramos una opción viable, el que el usuario esté realizando una rutina propia de otro día, ya sea de forma excepcional, o por un cambio en sus hábitos, como es que le cambien los días de rehabilitación. De esta forma, hay un riesgo, que consideramos bajo, y que asumimos ya que puede darse esta situación.

La segunda cuestión tiene que ver con la libertad del usuario de cambiar el camino que usa para ir a alguna parte, ya sea por una calle cortada por obras, o porque tiene que desviarse mínimamente a comprar el pan. Así que la monitorización no debería ser tan estricta como para valorar un solo camino para una misma rutina.

Teniendo estos dos conceptos claros, pasamos a explicar como funciona la monitorización.

Empecemos por el primer desplazamiento del día, en cuanto recogemos la primera muestra del histórico, actúa el monitor, lo primero que hace es rellenar una lista con todas las rutinas que parten del lugar de la muestra recién recogida e incrementamos el contador por primera vez.

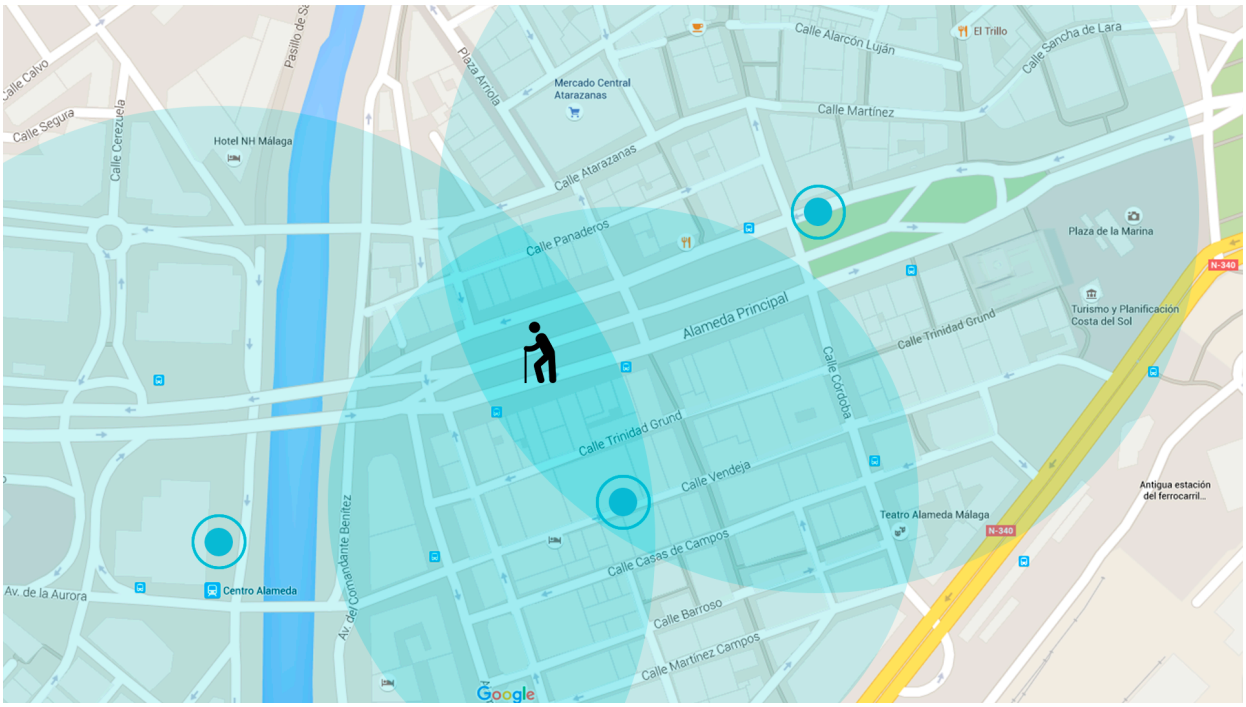

Figura 5. El usuario puede estar siguiendo cualquier rutina de la lista.

A partir de aquí, empieza el procedimiento descrito antes, comprobamos si han pasado los cinco minutos, si no han pasado, incrementamos contador; si sí que han pasado, vemos si el contador tiene un número de medidas aceptable, lo que significa que el usuario sigue moviéndose. Entonces pasamos a la parte central del procedimiento, ahora hay que comprobar si el paciente se ha desviado de alguna de las rutinas de la lista que recogimos antes, pero sin atender a caminos exactos. Lo que se hace es ir rutina a rutina de la lista, y se establece un radio de cierta holgura con centro en el destino de la rutina. El radio va a estar fijado en función del tiempo que quede para que el usuario llegue a su destino, teniendo en cuenta el tiempo que suele tardar en realizar dicha rutina. Así, a medida que va avanzando el tiempo, el radio va acotándose, haciéndose cada vez menor hasta que tenga que haber llegado a su destino. De esta forma, somos capaces de considerar que el usuario está en camino siempre que permanezca dentro de ese radio. En la figura 5 puede observarse la fase inicial de este procedimiento, cuando el usuario sale de un lugar y puede estar siguiendo cualquiera de las rutinas de la lista ya que está dentro de todos los radios con cierta holgura.

De todas las rutinas que habrá en la lista, la mayoría de las veces, habrá una que es la habitual, la que el usuario debería estar siguiendo ese día a esa hora, si en esta rutina específica se detecta, en alguna de las veces que se activa el monitor, que el paciente se ha desviado, se considera un riesgo bajo y se avisa al actuador para que realice su trabajo. En la figura 6, esta rutina concreta se corresponde con el área del punto rojo, indicando que ha quedado fuera del área permitida con el paso del tiempo y ya no se encuentra en su camino.

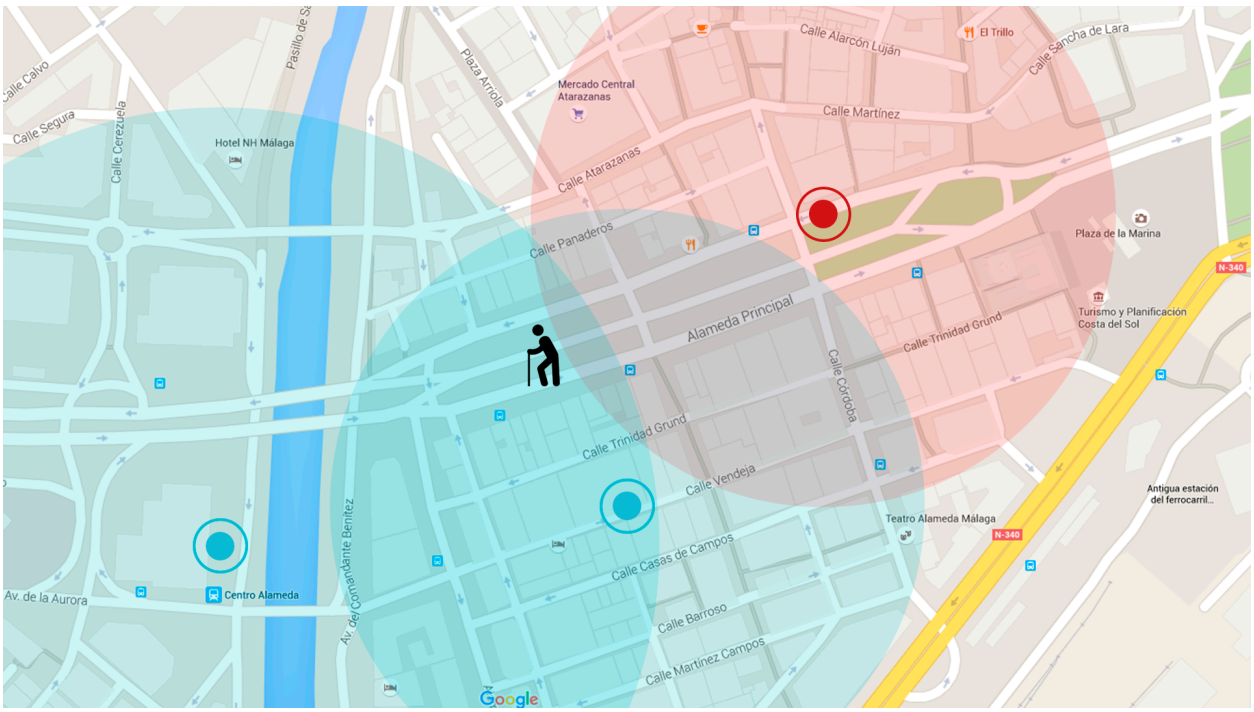

Figura 6. El usuario se desvía de la rutina que debería estar siguiendo.

El otro nivel de riesgo, el alto, se dará cuando en la lista no haya ninguna rutina que el usuario esté siguiendo, en cuyo caso se volverá a avisar al actuador pero para que esta vez, atienda a una alerta más grave. Esta es la situación descrita en la figura 7, el tiempo ha ido transcurriendo y las áreas permitidas han ido encogiendo hasta que finalmente se ha dado el caso de que el usuario no esta dentro de ninguna, por lo que no debe estar siguiendo ninguna de las rutinas habituales que salen desde su lugar de partida.

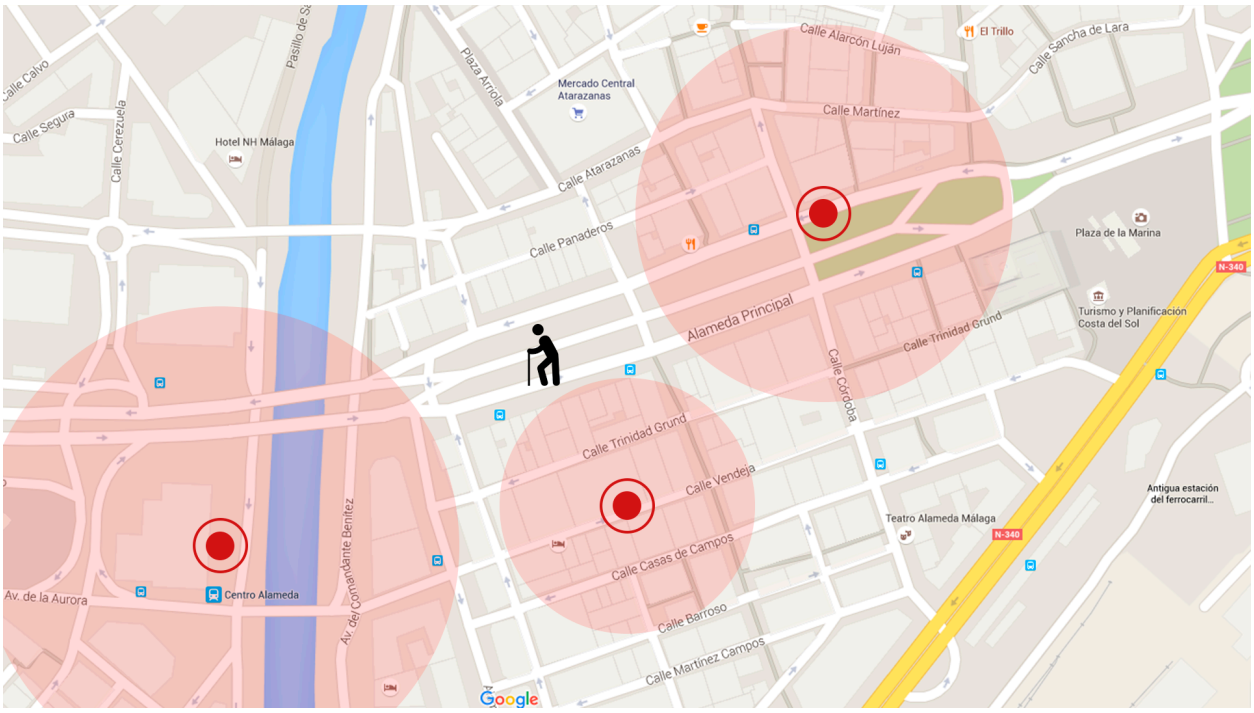

Figura 7. El usuario no sigue ninguna rutina.

El proceso es bastante intuitivo y fácil de comprender, si entre dos recogidas de muestras del histórico, han pasado mas de los cinco minutos, y no hay suficientes medidas, es decir, el usuario se ha parado, quiere decir que la muestra que acabamos de recoger, es cuando se ha vuelto a poner en movimiento, así que la usamos para resetear la lista de rutinas y empezar de nuevo el procedimiento. Aquí dejamos un extracto de pseudocódigo para ayudar al entendimiento del proceso.

```
private void monitor(Location loc) {
   //200 metros en 5 min
   if (horaActual - horaUltMonitor > 900000) { medidas = 0; }
   if (horaActual - horaUltMonitor > 300000) {
      if (medidas >= 6) {
         //Esta andando, Actualizo lista
         actualizarLista(loc);
      } else {
         //Se ha parado, Reseteo lista
         reseteaLista(loc);
      }
      //Actualizo la fecha
      horaUltMonitor = new Date();
   } else { medidas = medidas + 1; }
}
```

Otra función importante llevada a cabo por esta tarea es la de comunicación con el dispositivo del cuidador, para que este pueda llevar un seguimiento en tiempo real del paciente.

Para llevar a cabo esta comunicación hemos usado la plataforma de Nimbees. Esta se trata de una plataforma inicialmente pensada para fines publicitarios, sin embargo, incluye todas las herramientas que necesitamos para este cometido. Cabe destacar además, que Nimbees nos permite realizar este intercambio de mensajes de la forma más acorde posible a PeaaS, ya que la información sigue residiendo en el dispositivo móvil.

En cuanto a implementación, Nimbees es bastante fácil de usar. Para CareMe, no necesitamos ningún tipo de segmentación o filtros por localización, ya que los mensajes solo se intercambian entre dos usuarios registrados.

Los mensajes estándar de Nimbees simplemente transmiten un mensaje de texto y son automáticamente tratados por la aplicación receptora mostrando una notificación con el mensaje, aunque, estos no son el único tipo de mensajes que la plataforma ofrece. También ofrece unos mensajes personalizados, que son los que la app usa. Estos mensajes permiten enviar un fragmento JSON con los datos que queramos, en nuestro caso, necesitamos enviar las posiciones del paciente para que el cuidador pueda hacer el seguimiento en tiempo real, y son completamente gestionadas por el dispositivo receptor, lo que nos da libertad para usar este contenido como se necesite, y no en forma de notificación. A continuación mostramos un fragmento de código en el que se rellena este tipo de mensaje.

```
List<Double> coordenadas = new LinkedList<Double>();
coordenadas.add(loc.getLatitude());
coordenadas.add(loc.getLongitude());
NimbeesNotificationManager nimbeesNotificationManager =
      NimbeesClient.getNotificationManager();
Gson gson = new Gson();
nimbeesNotificationManager.sendNotification(gson.toJson(coordenadas),
      MessageContent.NotificationType.CUSTOM, filters, new
NimbeesCallback() {
   @Override
   public void onSuccess(Object arg0) {
      /*Éxito al enviar*/
   }
   @Override
   public void onFailure(NimbeesException arg0) {
      /*Error al enviar*/
   }
});
```

En la app CareMe-Caregiver tratamos este tipo de mensajes obteniendo las coordenadas del mismo, marcándolas a continuación en un mapa para visualizar la posición real del paciente.

Analizador

Se trata de la tarea con más cómputo del sistema, y por tanto, en la que más atención se ha tenido que prestar. La función que realiza es la de obtener las rutinas de movimiento del usuario a partir de los datos del histórico que la tarea del acumulador ha ido almacenando en la base de datos.

Así, esta tarea convierte lo que en un principio son muestras de latitud, longitud y fecha, en rutinas con una frecuencia, un lugar de partida y de llegada, y una convicción de fiabilidad de la misma. El proceso seguido para dicho análisis o aprendizaje, es el descrito por Davenport y Prusak [8], basado en la pirámide de conocimiento. Más adelante encontramos una sección de procedimiento de análisis en la que se relatará con más detalle el algoritmo implementado que usa esta tarea. *Una descripción más detallada del algoritmo implementado que usa esta tarea, se describe en la sección de procedimiento de análisis.*

Como hemos dicho, sin duda, se trata del proceso más costoso en cuanto a recursos del dispositivo debido a su elevada carga computacional, así que no podía tratarse de otro proceso como el de monitorización o acumulación que estuviera en ejecución constantemente. Esta tarea solo se realizará una vez al día y en un momento preciso. Este momento es habitualmente cuando el dispositivo entra en modo de carga, lo cual significa que ha sido enchufado a la corriente, y por consiguiente que no va a tener repercusión en la batería del mismo, además, cuando ponemos el teléfono a cargar, la mayoría de las veces no estaremos realizando una tarea con él, por lo que tampoco se produce una sobrecarga de cómputo en el mismo, siendo esto lo más cómodo para el usuario.

Hay situaciones que pueden darse en que el teléfono no se cargue en más de un día, en este caso, el análisis comenzará a una hora en la madrugada para reducir las molestias al mínimo posible. Sin embargo, para seguir garantizando el mínimo impacto sobre los recursos y la comodidad del usuario y su seguridad, este análisis a altas horas de la madrugada solo se realizará si el teléfono tiene un nivel de batería suficiente. Con esto tratamos de evitar así mismo, situaciones de riesgo que pudieran darse a dicha hora y nos interesa que la batería del teléfono sea afectada en lo mínimo

posible para que este pueda proseguir con la tarea de monitorización y que en ningún momento quede ilocalizable.

Otra tarea importante que realiza este componente es el borrado de datos antiguos ya utilizados para minimizar el uso de la memoria. En esta fase solo está implementado el borrado de datos con más de dos meses de antigüedad, sin embargo hay muchas medidas que pueden implantarse para futuras versiones. Una de las más interesantes es la de recuperación ante un fallo, podría hacerse cada cierto tiempo una copia de seguridad de los datos, o simplemente mantener algunos datos del histórico, los más significativos y que hayan sido usados, para poder volver a hacer el análisis y restaurar el sistema.

Actuador

Esta tarea se encuentra realmente implícita en el proceso de monitorización, ya que de lo que se trata es de mostrar los avisos al paciente y enviar las alertas pertinentes al cuidador cuando la monitorización detecte un desvío.

Como hemos dicho, podemos detectar distintos niveles de riesgo de desvío, como es el caso en que se salga de la rutina más probable, o que definitivamente, no esté siguiendo ninguna rutina. Según la definición de estos niveles de riesgo, el sistema se encarga de mostrar una notificación solo al paciente, o también mandar la alerta al cuidador. En esta fase de desarrollo, el comportamiento del sistema solo distingue riesgo bajo y alto, correspondientes a los tipos de salida de rutina mencionados. En situaciones de riesgo bajo solo muestra una notificación con vibración al paciente, así este puede darse cuenta y corregir el rumbo, o usar cualquiera de las opciones que tiene disponibles en la App, navegar hasta casa, o llamar a su cuidador o a emergencias. Cuando la situación es de alto riesgo, entonces esta tarea manda una alerta al cuidador para que este esté al tanto y pueda contactar con el paciente o con urgencias mientras monitoriza la posición de este en tiempo real.

En próximas versiones, el comportamiento de esta tarea debería ser configurable por el cuidador, dándole a elegir entre emisión de avisos a uno, a otro, o a los dos, según el nivel de riesgo.

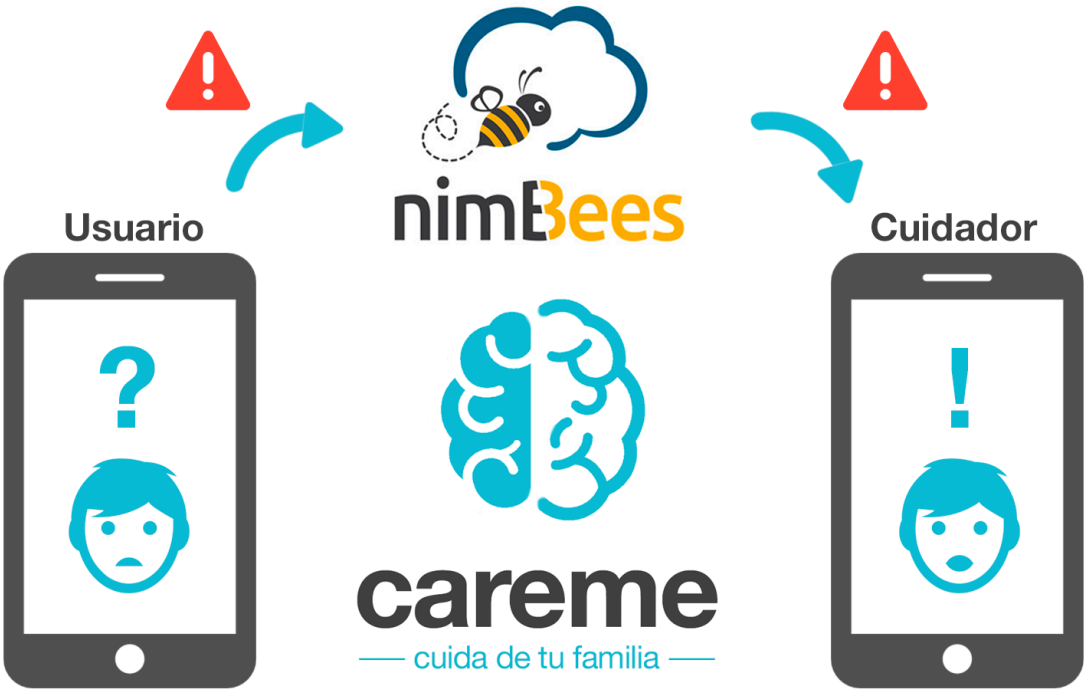

Figura 8. Diagrama de despliegue de las alertas con Nimbees.

Estos envíos de alerta de nuevo vuelven a realizarse mediante Nimbees. Solo que en este caso, cambia el tipo de mensaje con respecto al de monitorización. Esta vez cuando el cuidador lo recibe, muestra en su dispositivo una notificación con vibración para alertarlo de la situación de riesgo y que pueda reaccionar lo antes posible.

Al iniciar el sistema, se establecerá una lista de contactos de riesgo empezando por el responsable, así los avisos se podrán dar a más de una persona, sin embargo, al estar usando Nimbees para las comunicaciones, todas estas personas necesitaran la App del cuidador para poder recibir las notificaciones de forma gratuita sin el coste de un SMS.

Adicionalmente, en esta versión se ha desarrollado un servicio que diariamente va a encargarse de programar alertas para el paciente un poco antes de las horas de salida de cada una de ellas. El objetivo de estas alertas es el de recordar al paciente que debe prepararse para salir y acudir a su destino.

2.4 Estructura de Datos

Siguiendo las pautas dictadas por el modelo PeaaS, toda la información acerca de los usuarios y su contexto debería residir en su teléfono móvil, y no en ningún servidor ajeno, y es desde estos mismos teléfonos desde donde esta información es ofrecida al resto del mundo, ya sea a IoT o a otros usuarios.

Así nuestros dispositivos móviles, pulseras, gafas y todo este tipo de artilugios supondrían nuestra interfaz directa y automática con el mundo que nos rodea.

En CareMe los datos son almacenados en una base de datos SQLite en el propio teléfono. SQLite es un motor de bases de datos que a la par de ser transaccional y tener otras características interesantes como la no necesidad de servidor, o la poca configuración requerida, tiene un pequeño tamaño, lo que la hace ideal para este tipo de sistemas móviles. Además Android ya incorpora todas las herramientas necesarias para su manejo, proporcionando una API bastante completa.

Podemos distinguir entre cuatro tipos diferentes de datos según la descripción de la estructura del sistema: el acumulador necesita almacenar las medidas que vaya recogiendo a lo largo del día; el analizador usar estos datos para crear lo que serían las rutinas, que también deben ser almacenadas; el último tipo de datos serían los contactos de emergencia del paciente con su orden de prioridad y otras características. El último tipo será el dedicado a almacenar las citas puntuales que tenga el usuario, como el cumpleaños de un familiar o unos análisis.

Los dos primeros tipos, que denominaremos historial y rutinas, tendrán cierto parecido, sin embargo, es evidente que las rutinas además añaden información sobre la frecuencia, un porcentaje de convicción y modificará el campo de la hora por una hora de inicio y otra de finalización.

Así la base de datos se compondría de tres tablas principalmente. Sin embargo, son necesarias otras tablas de apoyo, tanto para completar la estructura de una rutina, como de apoyo para el análisis. En la figura 9 se muestra el diagrama entidad/relación de la base de datos final con todas las tablas y sus relaciones.

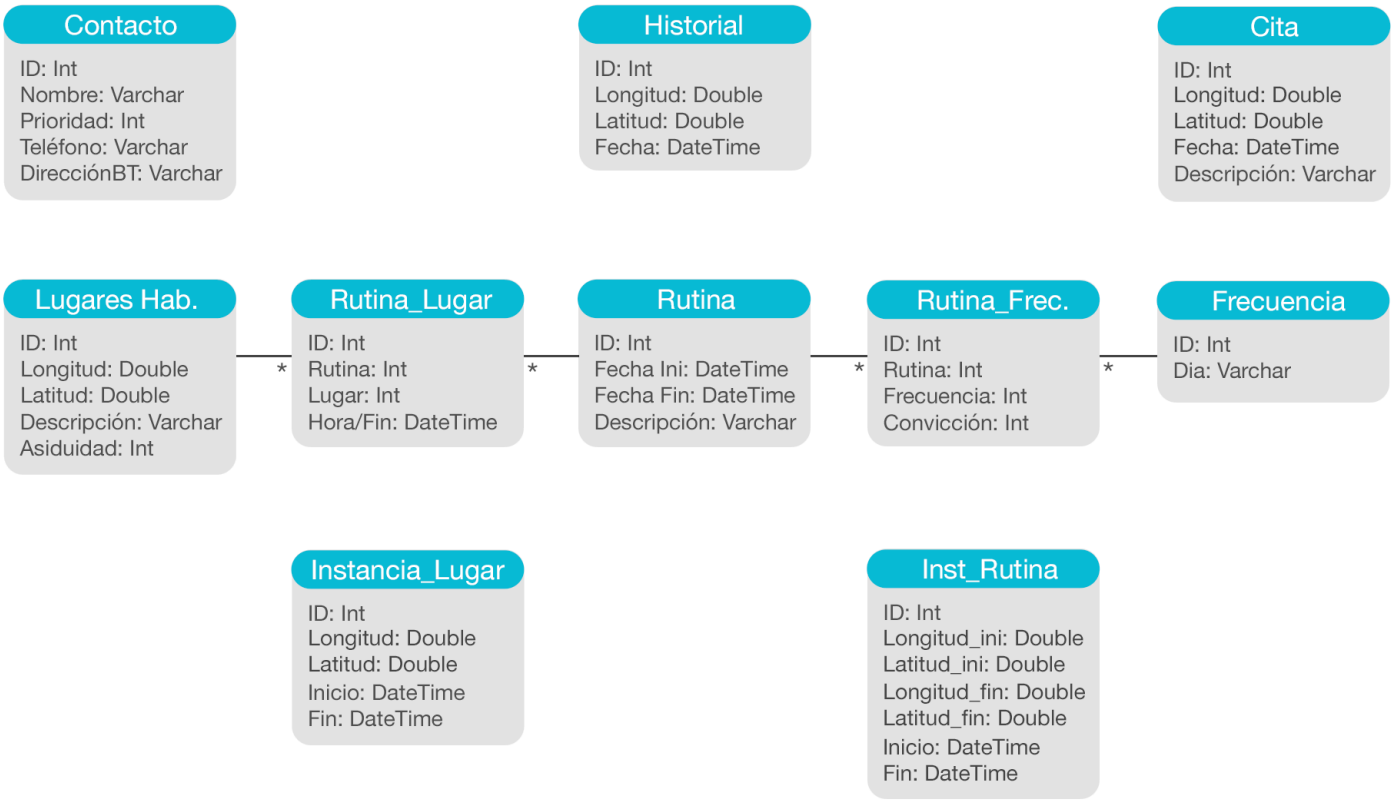

Figura 9. Diagrama entidad/relación de la base de datos del sistema.

Las primeras tablas serían las de los contactos, las citas y el historial que no tienen mayor relevancia.

Contacto

Esta tabla contiene la información de los contactos de emergencia del usuario.

- Nombre.
- Prioridad, es un entero del 1 al 3, siendo el 1 la mayor prioridad, la del responsable y el 3 la menor como bomberos…
- Teléfono.
- Dirección bluetooth, que nos puede ser útil para saber si el usuario en algún momento está acompañado por su responsable.

Cita

Aquí se recogen los eventos o citas eventuales que tenga el usuario como pueden ser citas con algún médico o para hacerse alguna analítica, cumpleaños de familiares… Contiene datos de:

- La fecha de la cita.
- La hora.
- El lugar en que tendrá lugar.
- Descripción de la cita.

Historial

Es donde el acumulador almacena todas las muestras que vaya recogiendo. Esta tabla tiene en primer lugar una columna ID que es un entero y representa la clave primaria. Tras esto tiene los datos que en principio necesitaríamos saber para que después el proceso analizador pueda establecer rutinas. Contiene datos de:

- Coordenadas GPS, que son dos números decimales largos que representarán latitud y longitud.
- Fecha, en que se ha tomado la muestra. Esta tiene un formato que incluye la hora para facilitar la definición de la rutina.

Se intentará borrar los valores más antiguos de esta tabla cada cierto tiempo, para que no llegue a ocupar demasiado en la memoria del teléfono del usuario en versiones futuras de la aplicación.

Además de estas tres tablas, la base de datos cuenta con la tabla Rutina, aunque esta también cuenta con otras tablas que se encuentran en una relación muchos a muchos con ella.

Estas tablas son las utilizadas para almacenar lugares conocidos y las frecuencias de las rutinas.

Rutina

Esta es la tabla fundamental del sistema, en ella se almacenan las rutinas y su grado de convicción y frecuencia. Todas sus columnas deberán inferirse de los datos de las muestras de la tabla Historial que tienen que ver con dicha rutina.

- Lugares a los que el usuario acude durante la rutina,

- Hora de inicio de la rutina.
- Hora de fin de la rutina.
- Frecuencia semanal con la que se repite la rutina (Lunes y martes, solo los miércoles…)

Los puntos a los que el usuario acude en la rutina se han modelado usando una relación muchos a muchos con una tabla llamada Lugares Habituales que contiene las coordenadas GPS de dichos puntos visitados por el usuario.

En cuanto a la columna frecuencia, también es una relación muchos a muchos con una tabla Día que contiene los nombres de los días de la semana; esta relación además tiene un campo convicción que será el porcentaje de convicción de que esa rutina se realice en ese día de la semana.

Lugares Habituales

Se trata de la tabla que se relaciona con las rutinas en la relación muchos a muchos de los lugares a los que irá acudiendo el usuario durante la misma. Representa los lugares en los que el usuario pasa más tiempo o acude con asiduidad, como su casa, el trabajo, la panadería…

Esta tabla contendrá las columnas necesarias para identificar estos lugares.

- Coordenadas GPS.
- Descripción del lugar, o anotaciones que se quieran hacer sobre él.
- Asiduidad, que representará un número entero, representando el más alto, el lugar donde más ha ido el usuario, que suele ser su casa.

Esta relación nos permite una muy fácil ampliación para versiones futuras en las que se podría considerar rutinas de mayor longitud con más lugares intermedios, y no de un lugar a otro únicamente, sin poder establecer relaciones entre ellos como en esta versión inicial.

Frecuencia

Es la tabla que está relacionada con Rutina, para representar la frecuencia semanal con la que se repite una rutina de desplazamiento del usuario. Solo contiene una fila por día de la semana de lunes a domingo. Para completar la información sobre una frecuencia semanal de una rutina, se añadió un campo de un entero para representar el

grado de convicción que se tiene sobre la repetición del desplazamientos ese día. Este campo se encuentra en la relación muchos a muchos entre las dos tablas.

Por último, es de esperar que en análisis, antes de obtener las rutinas completas, son necesarios una serie de pasos intermedios en los que los datos van adquiriendo cada vez una mayor complejidad. Estas tablas son, en orden ascendente de completitud, Instancia de Lugar e Instancia de Rutina.

Instancia de Lugar

Es el primer paso en el análisis. En esta tabla se intentan identificar los lugares en los que el usuario ha permanecido cierto tiempo. Es el paso previo a convertirse en un lugar habitual. Contiene:

- Coordenadas GPS, del lugar en cuestión.
- Fecha y hora de comienzo, de la permanencia en ese lugar.
- Fecha y hora del fin, de la permanencia en ese lugar.

Instancia de Rutina

Este es el último paso de los datos antes de convertirse en Rutinas. Estos datos ya tienen la misma forma que una rutina, a falta de la frecuencia.

- Coordenadas GPS, del lugar de partida.
- Coordenadas GPS, del lugar de llegada.
- Fecha y hora de salida, desde el lugar de partida.
- Fecha y hora de llegada, al destino.

2.5 Procedimiento de análisis

Según la filosofía PeaaS, los dispositivos móviles de los usuarios deben ser capaces de ser su interfaz con el mundo que los rodea, sin necesidad de que estos tengan que configurarlos. Estos dispositivos ya poseen toda la información que necesitan acerca de nosotros, y deben ser capaces de reaccionar ante estímulos teniendo en cuenta el contexto en el que se encuentra el usuario. Según PeaaS el dispositivo móvil del usuario es capaz de elaborar un perfil sociológico sensible al contexto de su usuario.

Para llegar a esta idea, es necesario tener conocimiento del usuario. La definición de conocimiento sigue siendo materia de discusión a día de hoy y, adopta diferentes formas dependiendo del contexto. En el contexto que atañe a este trabajo, una buena definición de conocimiento podría ser la proporcionada por Wright [12]. Esta hace énfasis en que el conocimiento, siempre estará fundado en hechos solidos.

> *"Knowledge signifies things known. Where there are no things known, there is no knowledge. Where there are no things to be known, there can be no knowledge. We have observed that every science, that is, every branch of knowledge, is compounded of certain facts, of which our sensations furnish the evidence. Where no such evidence is supplied, we are without data; we are without first premises; and when, without these, we attempt to build up a science, we do as those who raise edifices without foundations. And what do such builders construct? Castles in the air."*

Para entender esta definición, es necesario entender la diferencia entre hechos y conocimiento para obtener este último. Un buen punto de partida para establecer la relación entre los hechos y el conocimiento es la anteriormente descrita pirámide de DIKW, que divide el proceso de obtención de conocimiento en datos, información, conocimiento y sabiduría. Esta pirámide se muestra en la figura 10, donde pueden verse los niveles de ascensión hasta llegar a la sabiduría o inteligencia.

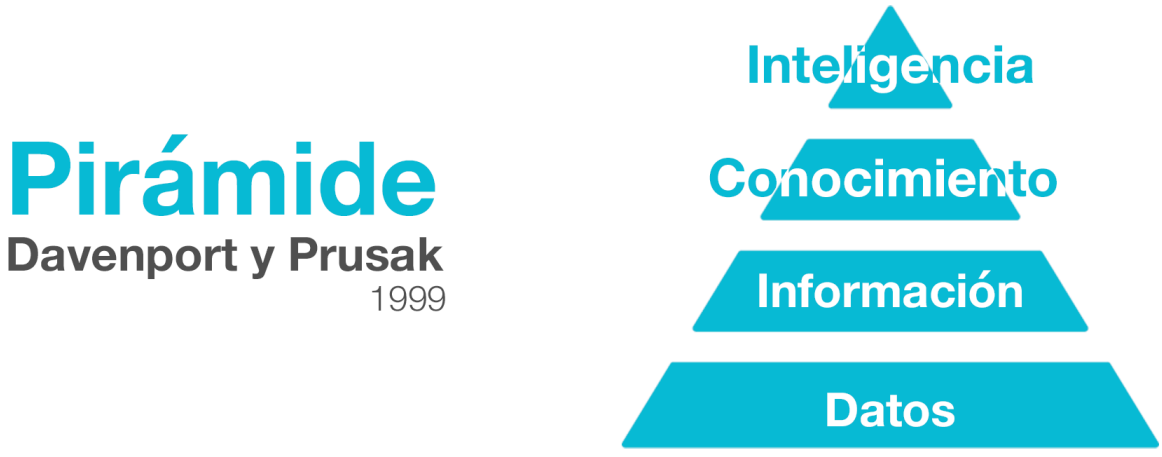

Figura 10. Pirámide del conocimiento de Davenport y Prusak.

Los datos en este contexto representan las evidencias observables, los estímulos que recogemos de la inspección del mundo. Esto es lo que representaría nuestro histórico de muestras.

La información tiene una naturaleza descriptiva, y es capaz de empezar a responder preguntas de quién, cuándo, qué… La información pueden ser datos útiles y con significado propio.

El conocimiento es una serie de reglas o expectativas que proporcionan un claro entendimiento de conjuntos de información. Es capaz de reconocer patrones en la información con experiencia e interpretación.

Por último, la sabiduría es el conocimiento que en un contexto puede utilizarse para generar nuevos conocimientos o parar tomar decisiones. Es lo que en nuestro trabajo serían las rutinas finales.

Siguiendo este modelo de proceso de obtención de conocimiento, el algoritmo diseñado e implementado para el análisis de rutinas de la aplicación CareMe va ascendiendo por la pirámide desde las muestras del histórico, hacia lo que se ha denominado instancias de lugar (información), de aquí al conocimiento de lo denominado instancias de rutina, y por último llegando a obtener las rutinas, que representarían la sabiduría.

Este algoritmo, debido a su elevado coste computacional, podría tener una fuerte repercusión en el dispositivo durante su ejecución. Es por esto, por lo que se decidió que este proceso solo se ejecutara una vez al día, y en las mejores condiciones posibles, para garantizar la comodidad del usuario. El algoritmo solo se ejecuta o bien, cuando se conecta el teléfono a la red eléctrica, o si esto no sucede un día, y este tiene aun un nivel considerable de batería, a altas horas de la madrugada de forma automática.

El procedimiento, como hemos dicho, sigue una serie de pasos hasta llegar a la obtención de las rutinas para ir ascendiendo por la pirámide del conocimiento.

1. Primer paso. Este paso parte desde la muestras recogidas por el proceso del acumulador durante el día. Recordamos que este recoge una muestra únicamente cuando el usuario se ha desplazado un cierto número de metros, así, si el paciente permanece en un mismo lugar, este margen de metros es considerado lo suficientemente grande para que mientras que el usuario camine por el interior de este, no se recoja ninguna muestra. De esta forma, solo tenemos muestras de los desplazamientos del usuario de un lugar a otro, pero no mientras está en ellos. En la

figura 11 puede observarse cómo se obtendrían las muestras según la situación del usuario.

Pues bien, teniendo esto claro, el primer paso del algoritmo consistirá en reconocer los lugares en los que ha estado el usuario y de que hora a que hora ha permanecido en ellos. Esta tarea resulta no muy complicada, consiste en ir tratando las muestras de dos en dos; si entre estas dos muestras ha pasado un tiempo coherente con el necesario en recorrer la distancia en metros que tiene como frecuencia el acumulador, entonces quiere decir que el usuario se esta moviendo, en caso contrario, tendríamos que la primera muestra es la llegada a un lugar donde ha permanecido un tiempo parado, y la segunda la salida de este. Así reconocemos las instancias de lugar. Es importante tener en cuenta que la primera muestra del análisis siempre hay que compararla con la última del análisis del día anterior, para no saltarse ningún lugar por medio. Cada vez que se crea una instancia de lugar, se compara con todas las demás almacenadas, ya que si se detecta que empieza a repetirse bastante, se crea un lugar habitual del usuario, al crearlo, puede darse el caso de que el lugar ya esté, por lo que en cuyo caso, solo incrementamos su campo de asiduidad. Gracias a esto es posible saber cual es la casa o el trabajo del usuario, ya que son los lugares con mayor asiduidad.

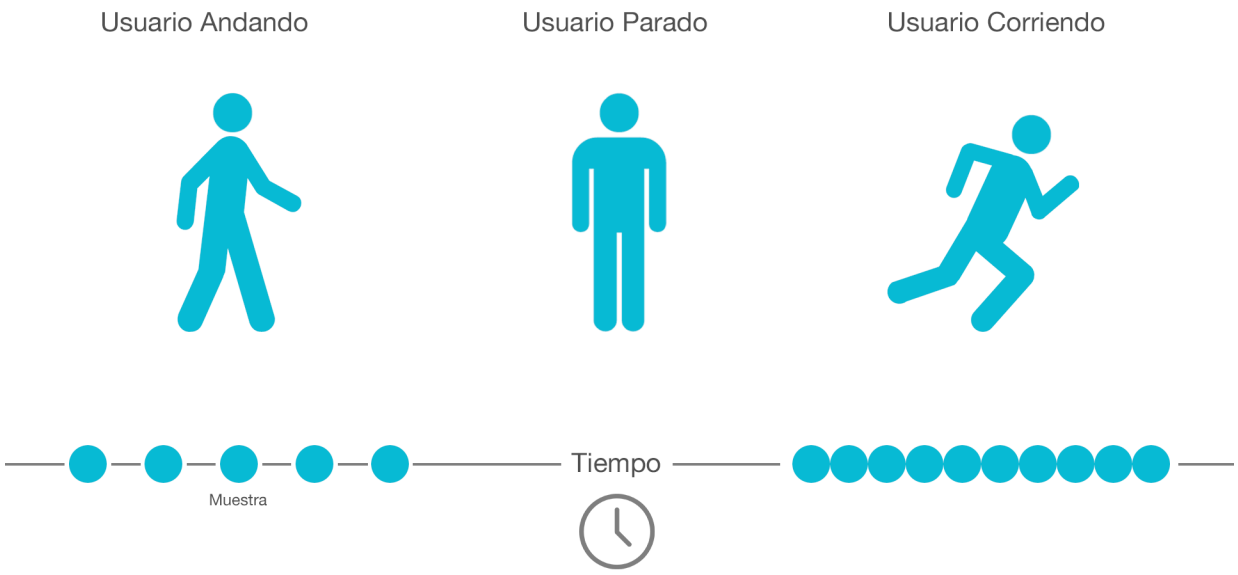

Figura 11. Recolección de muestras de posición a lo largo del tiempo.

2. Segundo paso. Ahora partimos de las instancias de lugar que tenemos en la base de datos. El procedimiento es muy parecido al anterior, solo que esta vez su significado es distinto. Una vez separados los lugares en los que ha estado el usuario en las instancias y conocidos sus momentos de llegada y salida, solo es necesario ir agrupándolos de dos en dos, de modo que si se tienen las instancias 1, 2 y 3, se obtendrían los grupos [1, 2] y [2, 3]. Estos grupos, como es de esperar, se corresponden con nuevas instancias de rutinas.

3. Tercer paso. En este último paso obtenemos las rutinas de desplazamiento del usuario. Para entender el procedimiento recordemos como se formaba una rutina en la base de datos.

   Este paso es prácticamente una prolongación del anterior. Cuando se almacena una instancia de rutina, se hace como con los lugares, se comprueba si hay alguna que empiece a repetirse. Si esto es así, cuando llegue a repetirse cierto número de veces, entonces pasamos a la creación de la rutina. Buscamos en la tabla de lugares habituales los lugares de la instancia y creamos la relación. Cabe destacar la fácil ampliación ya comentada del modelo a rutinas más complejas con mas paradas en lugares gracias a la relación y a su campo de hora de fin, que es importante saber para su posterior monitorización.

   Tras esto, se busca si la rutina ya existe, con esos dos mismos lugares de salida y llegada; si existe, se busca si se repite en la misma frecuencia, es decir, en el mismo día, si es así, incrementamos la convicción de la misma, en caso contrario, añadiríamos una nueva frecuencia a la rutina. Hay que tener en cuenta también, la hora a la que empieza la rutina para decidir si se repite en la misma frecuencia o no.

   Esta comprobación quedaría así.

```
l1.setLatitude(routineActual.getPlace().getLatitude());
l1.setLongitude(routineActual.getPlace().getLongitude());
l2.setLatitude(routineActual.getPlace().getLatitude());
l2.setLongitude(routineActual.getPlace().getLongitude());
l3.setLatitude(routineAlmacenada.getPlace().getLatitude());
l3.setLongitude(routineAlmacenada.getPlace().getLongitude());
l4.setLatitude(routineAlmacenada.getPlace().getLatitude());
l4.setLongitude(routineAlmacenada.getPlace().getLongitude());
if (l1.distanceTo(l3) < MIN_DISTANCE && l2.distanceTo(l4) <
   MIN_DISTANCE) {
   routineFreqs = db.getRoutineFreqs(routineAlmacenada.getId());
   for (int i = 0; i < routineFreqs.size() && !found; i++) {
      rf = routineFreqs.get(i);
      if (getWeekDay(routineActual.getStart()).equals(rf.getFrequency().getDay())) {
```

```
                if (getWeekDay(routineActual.getStart()).equals(rf.getStart())) {
                    //Actualizamos reliability
                    if (rf.getReliability() < 95) {
                        rf.setReliability(rf.getReliability() + 5);
                        db.insertRoutineFreq(rf.getRoutine().getId(),
                                rf.getFrequency().getId(), rf.getReliability(), rf.getId());
                    }
                    found = true;
                }
                }
            }
        }
```

2.6 Optimización de recursos

En el diseño de aplicaciones destinadas a dispositivos móviles, la optimización del consumo de recursos del dispositivo debe ser un factor director. Los recursos en los que se debe focalizar el esfuerzo son primordialmente la memoria del dispositivo, que suele ser uno de los principales problemas que los usuarios afrontan con su teléfono, la batería, y la capacidad de procesamiento del móvil (RAM y CPU), para no provocar que el dispositivo pueda quedarse bloqueado o disminuir mucho su rendimiento y velocidad durante su uso cotidiano.

En primer lugar, teniendo en cuenta el tamaño limitado de la memoria de los dispositivos, se ha optado por no guardar el historial de desplazamientos del paciente por un plazo indefinido. Como ya se ha mencionado anteriormente, una vez que los datos correspondientes al último día de actividad han sido procesados e integrados a la línea de tiempo del paciente podrían eliminarse. Otro caso es el de los datos intermedios generados en el análisis, podría ser útil mantener algunos y solo ir eliminando los más antiguos, ya que se trata de las muestras, pero ya tratadas y tendrán un menor impacto en la memoria. Gracias a esto puede implementarse una recuperación ante fallos en casos de que las rutinas se borren o estropeen para volver a generarlas.

Esto también podría resultar beneficioso cuando el paciente introduce un cambio en sus rutinas de comportamiento y desea regenerar su línea de tiempo a partir de las actividades realizadas en los últimos días. Por ejemplo, imagínese que el paciente cambia su domicilio habitual por una residencia con el objeto de conseguir asistencia durante todo el día.

Si ahora nos centramos en otro de los puntos críticos, la capacidad de procesamiento, el punto clave estará probablemente en el procesamiento de la actividad de la jornada, el reconocimiento de rutinas y su integración en la línea de tiempo, es decir, en el análisis, que es una tarea que consume muchos ciclos de reloj. Por ello esta actividad se lleva a cabo sólo en momentos en los que el Smartphone este siendo lo menos usado posible. Estos momentos son como hemos explicado, cuando se detecta que el teléfono está en carga, conectado a la red eléctrica, o en altas horas de la madrugada, en la que el usuario estará durmiendo.

Esta tarea, por extensión, también resulta crítica para el ultimo recurso a tener en cuenta del dispositivo, la batería. Es por esto por lo que se decidió dar prioridad entre estas dos opciones a la de mientras esté cargando, ya que la ejecución de la tarea en este momento, no tendría repercusión en la duración de la batería.

Por último, todas las actividades de registro de localización física y monitorización también consumen mucha batería y capacidad de procesamiento durante el día. Sin embargo, en este caso resulta difícilmente evitable. Si se desea mucha precisión en los registros de información y en los cálculos de cercanía se requieren cadencias de registro muy elevadas. No obstante, como ha podido comprobarse, existen gran número de parámetros configurables en la aplicación con el objeto de poder adecuar estas cadencias a las características de cada paciente.

Además, se han implementado medidas ya explicadas en sus respectivos procesos, por ejemplo en el de monitorización, para reducir el cómputo, sobre todo en frecuencia, sin llegar a observar una pérdida de precisión en la monitorización muy grave, ya que en la versión actual, se detectarían como máximo en 5 minutos.

Actualmente se dispone de un prototipo funcional de la aplicación al que corresponden las imágenes mostradas. Dicho prototipo monitoriza la actividad del paciente, aprende sus rutinas en base a puntos de inactividad y rutas entre ellos y realiza detecciones básicas de si el paciente se encuentra en rutina o no. Sin embargo, todavía existen múltiples extensiones que pueden mejorar las medidas de optimización de recursos, como por ejemplo detectar la conexión a puntos WIFI, permitiendo la desactivación de algunos procesos durante un tiempo o la presencia de un acompañante del paciente, quizás mediante Bluetooth, que permitiría también reducir el computo de monitorización.

3. Proceso de desarrollo

Para la validación del modelo PeaaS se decidió realizar el desarrollo de una aplicación móvil, que es el dispositivo en el que se centra el modelo.

Antes de empezar a implementar, lo primero que se hizo fue una búsqueda a fondo sobre documentación del modelo y pensar en una idea actual que se amolde bien al modelo y a la que este aporte una mejora significativa. Es así como se decidió implementar CareMe, una aplicación para dispositivos móviles en el sector de la salud, donde este modelo, que en definitiva pone a las personas en el centro de importancia, encaja a la perfección.

El siguiente paso fue la elaboración de los requisitos funcionales y no funcionales que la aplicación debería incluir para suponer una prueba firme de la validez del modelo. Es en este momento donde se toman decisiones como en que plataforma se implementaría, que finalmente ha sido Android, debido a su fácil accesibilidad por todo el mundo.

Lo último que restaba antes de empezar a implementar era, cómo íbamos a implementarlo, CareMe suponía una aplicación de tamaño considerable donde se realizaban múltiples tareas que a priori tenían que ser simultáneas. Se diseñó la arquitectura del modelo de forma parecida a la que hemos explicado antes, sin embargo, ya se pudo tener una idea de cómo sería el sistema y que componentes eran los necesarios para su correcto funcionamiento.

Con toda la arquitectura ya decidida y demás estudios previos realizados, se empezó a implementar CareMe, la app que se decidió que debería ser la primera en implementarse fue evidentemente la del paciente, debido a que es en esta donde esta el 90% del computo de la aplicación, además de ser el corazón del sistema.

Lo primero en implementarse fue la estructura de datos, que consiste en una base de datos SQLite como ya hemos explicado, aunque en esta primera fase, la base de datos no quedó completa, aún no se sabía que datos intermedios iba a generar el análisis y si realmente sería necesario conservarlos.

Después, se decidió empezar por la parte con más cómputo y a priori más complicada del sistema, que es el análisis. En un principio, solo se implementó la recogida de muestras del GPS del Smartphone del usuario para así observarlas y decidir el siguiente paso. Al comprobar la forma en que podíamos recoger las muestras, solo cuando el usuario se desplazara una cierta distancia, fue fácil avanzar al siguiente paso. Como se ha explicado, al recoger muestras solo durante los desplazamientos, mientras

el usuario permanece en el mismo lugar, no se recoge ninguna muestra. Así que pusimos una distancia acorde para que el usuario siempre que estuviera en un mismo lugar, no recogiera muestras, aunque se mueva por el interior de este. Esta distancia ha ido variando a lo largo de toda la implementación, ya que igual que se seguía desarrollando, se iba probando lo que ya se tenía para ir ajustando estos “números mágicos” de forma que la recolección de datos sea óptima. Así que lo primero que conseguimos obtener, analizando las muestras que recogíamos, fueron los lugares donde permanecía el usuario durante un tiempo, que eran las muestras consecutivas que tenían una distancia temporal entre ellas considerables. Estas son las Instancias de Lugares.

Como el análisis sabíamos que es el proceso de más computo, se decidió que se realizara cuando el teléfono estuviera menos en uso, además que supusiera el mínimo gasto de batería. Así que tras buscar información sobre tipos de procesos en Android, se encontró el que mejor cuadraba con el caso, un BroadcastListener. Este componente Android nos permite estar atentos a llamadas del sistema y actuar en consecuencia; la llamada en consecuencia que activa este sistema es la de conexión a la red eléctrica para cargarse. De este modo, al poner el móvil a cargar por la noche antes de dormir, al la mañana siguiente se dispone de una lista de los lugares donde se ha estado el día anterior.

Una vez se tuvo los lugares a los que el usuario acudía, el siguiente paso sería unirlos, para construir los desplazamientos pasados. Esta tarea no resultó muy difícil ya que durante la obtención de los lugares, también era posible obtener la hora a la que llegaba a este, que era la de la primera muestra, y a la hora a la que salía, que era la de la segunda muestra de la comparación. Así con estos datos, solo con listar los lugares en orden de fecha y hora, y con unirlos dos a dos, se obtienen todos los desplazamientos que el usuario realiza.

El último paso del análisis fue la obtención de las rutinas finales y los lugares habituales.

Esta versión estuvo en un periodo de prueba para ir ajustando los parámetros y corrigiendo algunos errores para que el análisis consiguiera realizarse de forma correcta. Uno de los principales problemas encontrados fueron los fallos del GPS en los que se tomaba una muestra aunque no se hubiera producido ningún movimiento.

Esta aplicación se muestra en la figura 12, en la cual podemos apreciar que solo contaba con algunos botones para realizar lecturas de la base de datos.

Figura 12. Versión de prueba de CareMe.

El siguiente paso fue la monitorización. Esta como ya se ha explicado debe hacerse simultáneamente con la acumulación de histórico. Esto fue lo que decantó que ambos procesos estuvieran dentro de un servicio de Android, así siempre permanecería activo, aunque se cerrara la ventana de la aplicación.

Antes de empezar a implementar hubo que tomar decisiones acerca de cómo se iba a abordar el problema, para no forzar al usuario a realizar siempre el mismo camino, solución que además hubiera obligado a almacenar además puntos intermedios de los caminos, lo cual no es muy eficiente para la memoria del dispositivo. Finalmente se diseñó el algoritmo de monitorización ya explicado. Con este algoritmo no fue necesario corregir muchos errores.

Para poder hacer las pruebas, llevando un mejor seguimiento, se desarrolló una funcionalidad para hacer un volcado de la base de datos a un archivo de texto, y así observar la progresión de los datos. Esta funcionalidad aun se encuentra algo oculta en la fase actual de la aplicación por si se detecta algún problema.

Mientras se ponía a prueba lo del funcionamiento de CareMe se diseñaron la interfaz y algunas funcionalidades extra, como la configuración de alarmas para media hora antes de que empiece cada rutina, a modo de recordatorio para el paciente.

Con todo esto listo, solo quedaba diseñar una app sencilla para el cuidador, que le permitiera realizar un seguimiento de la posición del usuario y recibir notificaciones de

alerta en caso de peligro. Esta app consiste en un mapa donde se va mostrando la situación del paciente en todo momento y dos botones para llamar al paciente y al 112 en caso de emergencia.

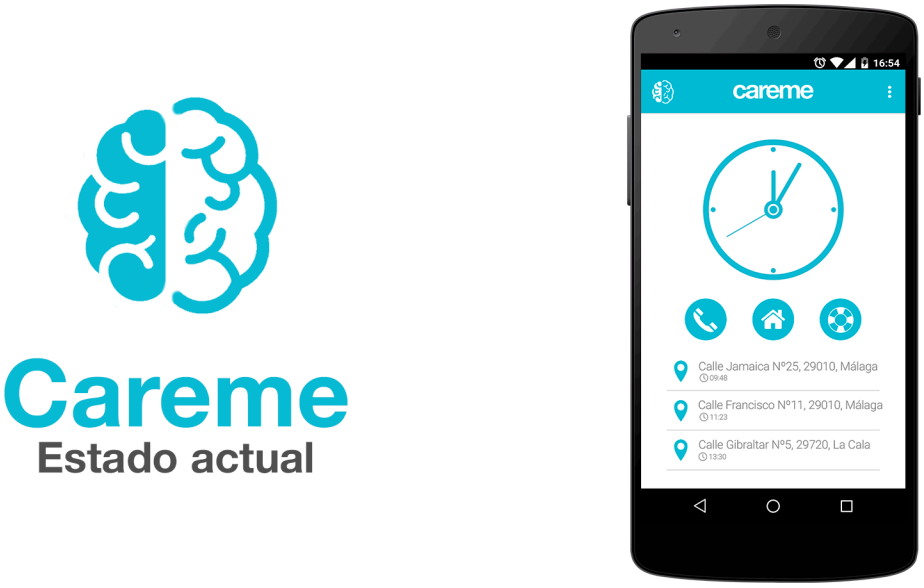

Figura 13. Versión actual de CareMe.

Esta comunicación, tanto de alertas como de seguimiento se realizó con el uso de la plataforma Nimbees para envío de mensajes push. Para la configuración de esta plataforma seguimos los pasos proporcionados por la documentación de la plataforma y algunos consejos del equipo creador de la misma. Una ventaja de Nimbees es la capacidad que otorga de monitorización de todos los mensajes mediante su portal web, que puede verse en la figura 14.

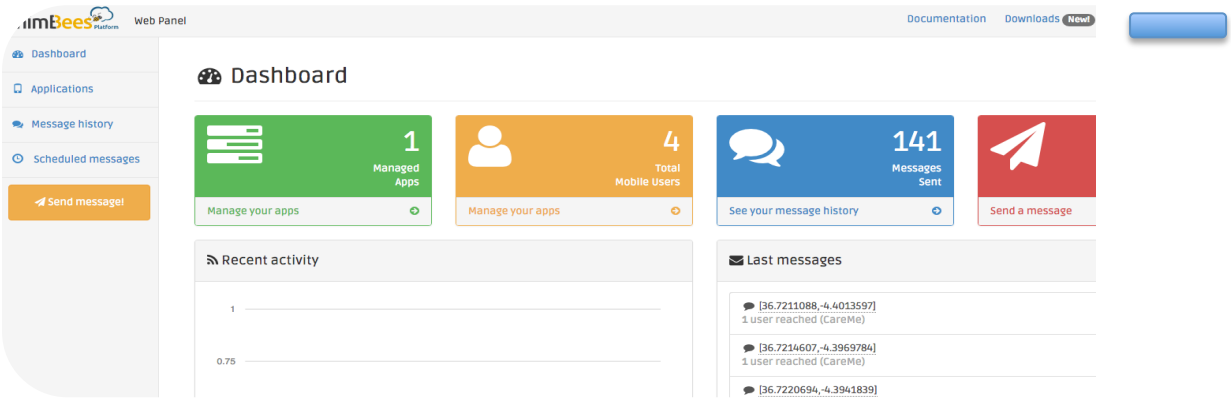

Figura 14. Panel web de Nimbees.

Para esta fase de desarrollo se escogió el plan desarrollador de Nimbees que es gratuito, sin embargo limita el número de mensajes que se pueden enviar.

Este es el estado actual del sistema, sin embargo, está en una fase muy temprana, lo que significa que aún pueden introducirse muchas nuevas funcionalidades y mejoras en las versiones futuras. Algunas de estas funcionalidades pueden ser por ejemplo el seguimiento bajo demanda, para reducir el envío de mensajes a cuando el cuidador lo solicite; y todas las funcionalidades descritas en los requisitos acerca de la configuración de CareMe y acciones que debería poder realizar el cuidador, como la creación de rutinas y citas.

Finalmente el trabajo se concluye con la elaboración de esta memoria. Esta elaboración se lleva a cabo gracias a datos recabados en las investigaciones realizadas, experiencias personales y anotaciones, entre otras cosas que se han ido recogiendo a lo largo de toda la implementación de la aplicación con el objetivo de facilitar mucho más su escritura y no saltarse ningún detalle para que este trabajo sirva como principio para otros futuros.

4. Trabajos relacionados

PeaaS es un modelo que persigue poner en valor el contexto de las personas proporcionándolo como un servicio desde el teléfono móvil. En los últimos meses, hemos podido observar como las grandes compañías, y en general la tecnología, está tratando de encontrar el modo en que nuestra interacción con el resto de personas o cosas presentes en nuestra vida sea mucho más sencilla, podríamos decir que automática. Y es sin duda esto lo que persigue este modelo, dando a este trabajo una especial relevancia además de suponer una gran cantidad de aportaciones a esta materia.

Algunas de las tecnologías más avanzadas en este campo por el momento, como avecinamos, son las aportadas por las grandes empresas. Es el caso de Siri[6] de Apple, Cortana[7] de Microsoft o Google Now[8]; Aunque también hay proyectos similares que no vienen de ellas, como es Sherpa[9]. Todas estas ofrecen una funcionalidad parecida, proporcionando al Smartphone capacidades atribuibles a personas, y encargando a este funciones de asistencia en nuestro día a día especial y adaptada para el usuario. En las últimas versiones de los sistemas operativos móviles de estas compañías, puede ir observándose esta tendencia, dedicando a este propósito una pantalla de escritorio, o algunas etiquetas en el área de notificaciones. Estos avances suelen ser del tipo de detectar cosas de interés para el usuario, como las noticias con más repercusión alrededor de este, que entren dentro de sus secciones de interés. O los restaurantes del alrededor mejor valorados que ofrezcan el tipo de comida más habitual del usuario.

La aplicación de PeaaS en este trabajo consiste en poner el contexto del *Paciente* al servicio del *Cuidador.* Ya existen otras aplicaciones móviles basadas en una arquitectura convencional que asisten a pacientes de alzhéimer con el mismo objetivo que CareMe. Una de las pioneras en este género es Tweri. Esta aplicación proporciona funcionalidades básicas para identificar las zonas donde el paciente se encuentra seguro de forma que se notifiquen alertas al cuidador si las abandona. El cuidador puede seguir al paciente a través de una aplicación diferente. Además, si el paciente se encuentra inseguro puede lanzar una alerta al cuidador mediante una pulsación en la interfaz. Con funciones mejoradas y el mismo objetivo, Cerqana, cuyo panel web puede verse en la figura 15, permite la fijación manual de zonas seguras y peligrosas

[6] Siri. https://www.apple.com/es/ios/siri/
[7] Cortana. http://www.windowsphone.com/es-es/how-to/wp8/cortana/meet-cortana
[8] Google Now. https://www.google.com/landing/now/
[9] Sherpa. http://sher.pa/

en un mapa, aviso vía notificación si entra en alguna zona marcada como insegura, fijación manual de rutas habituales del usuario o detección de caídas o desvanecimientos.

Figura 15. Panel web de Cerqana.

En comparación con CareMe, ambas aplicaciones adoptan una estrategia en la que el dispositivo móvil del paciente se encarga de subir su localización a un servidor con una cadencia determinada. En el servidor se realizan todos los análisis de datos y el tratamiento de alertas. Ninguna de ellas aborda la detección y aprendizaje de las rutinas del paciente, ni la actuación en tiempo real ante alertas, sino sólo basada en los datos que se encuentran en el servidor y que normalmente tienen minutos de retraso que, en algunos casos, pueden resultar decisivos. Esta característica es difícilmente solventable mediante el uso de arquitecturas convencionales ya que implicaría cadencias muy altas en la subida de datos al servidor con el consiguiente consumo de batería. Plataformas como UrbanAirship[10] advierten que cadencias de registro y subida de datos cercanas a los 30 segundos implican el consumo de la batería del dispositivo en un periodo de 30 a 40 minutos.

Para la elaboración de este trabajo, finalmente se ha optado por emplear un algoritmo propio y muy básico para el análisis y aprendizaje de rutinas. Sin embargo, el uso de

[10] UrbanAirship. http://urbanairship.com/

herramientas como RDFQuery [11], RDFStoreJS [12], Nools [13] o AndroJena [14] podrían contribuir a mejorar nuestra propuesta. Su uso está contemplado en un siguiente paso cuando se proceda a incorporar datos al análisis más allá de la localización.

[11] RDFQuery. https://code.google.com/p/rdfquery/
[12] RDFStoreJS. https://github.com/antoniogarrote/rdfstore-js
[13] Nools. https://github.com/C2FO/nools
[14] AndroJena. https://github.com/lencinhaus/androjena

5. Conclusiones

El progresivo desarrollo de IoT y la salida al mercado de nuevos dispositivos *wearable* están haciendo cambiar la forma en que interactuamos con nuestro entorno, favoreciendo el surgimiento de escenarios cada vez más interconectados y de complejidad creciente. Sin embargo, el estado actual de la tecnología, obliga a una continua intervención del usuario. Modelos como PeaaS abogan por el uso del teléfono móvil como interfaz del usuario con su entorno, y permiten el desarrollo de perfiles sociológicos que pueden ser ofrecidos como servicios a terceros de forma controlada, favoreciendo la transición de IoT a IoP.

En este trabajo presentamos el desarrollo de una aplicación para la supervisión y seguimiento de enfermos de alzhéimer, como prueba de concepto del modelo PeaaS. A partir de la monitorización de la señal GPS del teléfono, la aplicación desarrollada es capaz de aprender las rutinas de desplazamiento del usuario y de tomar decisiones en caso de desviaciones de las mismas que puedan implicar un peligro para el enfermo. CareMe ofrece una funcionalidad significativamente superior a otras similares actualmente en el mercado, y muestra cómo los conceptos inherentes al modelo PeaaS pueden ser puestos en práctica en un escenario de uso real, más allá del caso de estudio elegido como prueba de concepto. Actualmente nos encontramos en una fase alpha del desarrollo de la aplicación, aunque ya es plenamente funcional, sin embargo, como hemos repetido numerosas veces en este articulo, aún admite numerosas funcionalidades más y mejoras a realizar para nuevas versiones más avanzadas y que pueden llevar CareMe a su uso real en la vida cotidiana de estas personas que lo necesitan.

5.1 Trabajos Futuros

En cuanto a trabajos futuros, cabe señalar la incorporación progresiva de nuevos algoritmos de aprendizaje, para lograr una mayor precisión tanto en la detección de rutinas de desplazamiento, como en la predicción de las mismas y en la detección de desviaciones respecto al comportamiento habitual. La integración con servicios como Google Maps permitiría obtener información de interés sobre el entorno de las coordenadas del usuario, afinar la detección de situaciones peligrosas y proporcionar ayuda al usuario para retomar su rutina de desplazamiento en caso de que se haya extraviado. Por otro lado, el uso de la señal *wifi*, y *bluetooth* del teléfono permitiría un

mejor seguimiento de la actividad del usuario en interiores, por ejemplo mediante iBeacons[15], y la detección de si está acompañado por sus cuidadores o no.

Mediante esta prueba de concepto se ha demostrado las numerosas aportaciones que puede realizar el modelo PeaaS a la tecnología actual, convirtiendo la vida de los usuarios en una vida mucho mas cómoda y permitiendo la integración de estos usuarios en el mundo que los rodea sin ningún esfuerzo y de forma automática.

El modelo supone un gran adelanto para muchas tecnologías, entre otras IoT, permitiéndole adaptarse automáticamente a cada usuario Esta idea abre la puerta a multitud de nuevos escenarios.

- Nuevos protocolos de pago. No es necesario tener la información de pago en el móvil. Únicamente cuando se esté pagando, la entidad financiera desplegará un servicio de pago seguro en tu Smartphone.
- Logística de recolección de residuos. El teléfono centraliza información acerca de la cantidad y tipos de residuos que generamos en casa, así preguntando al usuario y sus vecinos se podrá planificar una recogida más eficiente. También podrá avisar del momento en el que dejar la basura en el contenedor.
- Política. El Smartphone podría plantear encuestas a sus usuarios de forma que el gobierno sería luego capaz de recoger los resultados agrupados por edad, sexo… Esta información siempre quedará segura, ya que en ningún momento sale del teléfono.
- Planes energéticos. El dispositivo móvil podrá elaborar planes energéticos más eficientes para cada usuario.
- El caso que ocupa la prueba de concepto, ayuda a personas mayores o enfermos. Estos no tendrán que enterarse de nada ni saber nada acerca de tecnología, la aplicación se despliega en sus Smartphone y la ofrece a sus cuidadores y a ellos mismos para mejorar sus hábitos.
- Estrategias de marketing avanzadas. Se podría extraer información directamente de los clientes, explorar sus necesidades. Así se programarían las estrategias teniendo toda esta información en cuenta.
- Salud. Los teléfonos móviles ofrecerían datos acercar de parámetros de salud. Analizar las enfermedades de los usuarios y ayudarles con la medicación que

[15] iBeacons. https://developer.apple.com/ibeacon/

usan. Además obtendrían estadísticas de salud acerca de distintos tipos de enfermedades por áreas geográficas.

6. Referencias

[1] Y. Cui, M. Honkala. “A novel mobile device user interface with integrated social networking services”: International Journal of Human-Computer Studies, 71(9), 919-932, 2013.

[2] M. Bakardjieva, R. Smith. “The Internet in everyday life computer networking from the standpoint of the domestic user”: New Media & Society, 3(1):67-83, 2001.

[3] A. Dohr, R. Modre-Opsrian, M. Drobics, D. Hayn, G. Schreier. “The Internet of Things for ambient assisted living”: Information Technology - New Generations (ITNG), Seventh International Conference on, pp. 804-809, IEEE Computer Society, 2010.

[4] L. Sha, S. Gopalakrishnan, X. Liu, Q. Wang. “Cyber-physical systems: A new frontier”: Machine Learning in Cyber Trust, pp. 3-13, Springer, 2009.

[5] J. Guillen, J. Miranda, J. Berrocal, J. Garcia-Alonso, J.M. Murillo, C. Canal. “People as a Service: A Mobile-centric Model for Providing Collective Sociological Profiles”: Software, IEEE 31(2), 48-59, 2014.

[6] L. Srivastava “Mobile phones and the evolution of social behavior”: Behaviour & Information Technology, 24(2), 111-129, 2005.

[7] A. Oulasvirta, T. Rattenbury, L. Ma, E. Raita “Habits make smartphone use more pervasive”: Personal and Ubiquitous Computing, 16(1), 105-114, 2012.

[8] T.H. Davenport, L. Prusak. Working Knowledge : How Organizations Manage What They Know. Harvard Business School Press, 1998.

[9] J. Miranda, N. Mäkitalo, J. Garcia-Alonso, J. Berrocal, T. Mikkonen, C. Canal, J.M. Murillo. “From the Internet of Things to the Internet of People”, Internet Computing Magazine, IEEE, 19(2), 40-47, 2015.

[10] N. Mäkitalo, J.Pääkkö, M. Raatikanien, V. Myllärniemi, T. Aaltonen, T. Leppänen, T. Mänistö, t. Mikkonen, “Social Devices: Collaborative Co-located Interactions in a Mobile Cloud” Proceedings of the 11th International Conference on Mobile and ubicuous Multimedia, MUM, 2012.

[11] N. Mäkitalo. Building and programming ubiquitous social devices, Proceedings of the 12th ACM international symposium on mobility management and wireless access (MobiWac ’14), pp. 99-108, ACM 2014.

[12] Frances Wright. *Course of Popular Lectures*. Office of the Free Enquirer, 1829. 24

Anexos

Anexo I

Manual de Usuario

La aplicación CareMe es una aplicación muy fácil de usar para el usuario, lo que es para su propósito esencial, ya que un gran porcentaje de sus usuarios serán personas mayores con una edad superior a los 60 años.
Para el afectado, la aplicación debe actuar sin que él si quiera se de cuenta, dejándole solo realizar acciones de emergencia y consulta. No depende de él activar o desactivar algún mecanismo ni deberá introducir ningún dato.
En la fase actual de desarrollo la aplicación del paciente solo tiene una pantalla.

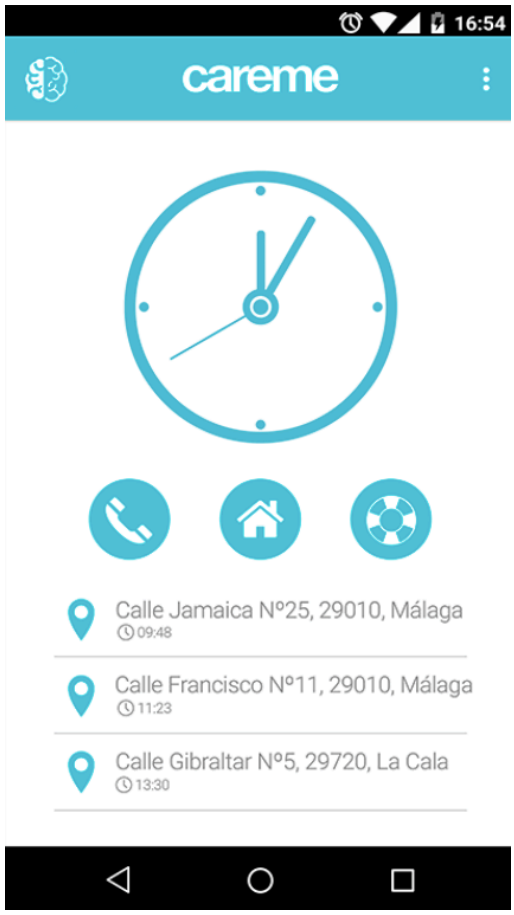

Figura 1. Aplicación paciente.

En ella encontramos una serie de botones y componentes muy intuitivos.

El primero y más grande de todos es un reloj, este no tiene más comportamiento que mostrar la hora para que el usuario sea capaz de ver de forma rápida lo que le queda para la siguiente rutina.

Debajo del reloj nos encontramos con tres botones, los tres realizan acciones a realizar en caso de que el usuario se pierda.

Sirve para llamar al cuidador.

Sirve para iniciar una navegación hasta la casa del paciente, así si se ha perdido, este podrá volver a casa sin problema.

Sirve para llamar al número de emergencias, como actualmente la aplicación se encuentra en una versión final comercial, aún la aplicación no llama directamente, únicamente marca el número.

Por último, debajo de estos botones, encontramos una lista. Esta lista se rellena con las rutinas que tiene previstas el paciente para el día en curso junto con la hora a la que tiene que salir para llegar al destino.

Si el usuario pulsa sobre cualquiera de los ítems de la lista, será redirigido a un mapa con una marca en el lugar especificado en el ítem.

En el menú contextual de la esquina arriba a la derecha nos encontramos con dos opciones que en un principio solo estarán disponibles en estas primeras versiones. La primera de ellas se llama *Habilitar alarmas*, esta pone en funcionamiento la funcionalidad de emitir una alarma en el teléfono media hora antes de empezar cada rutina, a modo de recordatorio para el usuario. La segunda es *Imprimir datos*, esta es una funcionalidad de testing, realiza un volcado de la base de datos en un fichero de texto, así podemos analizar el comportamiento del algoritmo de análisis.

La segunda aplicación es la del cuidador, esta aplicación tendrá muchos más comportamientos en el futuro. En la fase actual, la aplicación consiste de un mapa en el cual se muestra en todo momento la posición del paciente, y dos botones flotantes en la parte inferior derecha de la pantalla.
Estos dos botones tienen el mismo comportamiento que los que tienen el mismo icono en la aplicación del paciente, llamar al paciente y a emergencias.

Ambas aplicaciones muestran una pantalla la primera vez que se abren en la cual se deben rellenar el número de teléfono y el email del compañero respectivo de cada uno, es decir, el paciente los del cuidador y viceversa. Esto sirve para configurar la comunicación entre ambas aplicaciones y el registro de los usuarios en Nimbees.

Anexo II

Documentación técnica

A continuación se darán detalles técnicos de las principales funciones desarrolladas para la versión actual de CareMe. Empezaremos por la aplicación del cuidador, que es la más simple, y luego entraremos a exponer la del paciente.

CareMe-Caregiver

En esta aplicación solo encontramos una clase de interés, es la encargada de recibir los mensajes personalizados de Nimbees. Los mensajes normales son tratados automáticamente.

CustomNotificationManager

Es la clase donde se realiza el tratamiento de las *Custom notifications* de Nimbees. Para poder realizar su comportamiento es necesario que extienda *NimbeesNotificationManager* e implemente el método de tratamiento de este tipo de notificación.

Custom notifications

Las notificaciones personalizadas se gestionan mediante el método handleCustomMessage.

```
@Override
public void handleCustomMessage(long idNotification, String content, Map<String, String>
additionalContent) {
   // Insert your code here
}
```

La gestión de los mensajes se deja por completo al desarrollador, en nuestro caso, almacenamos las coordenadas que nos envía la aplicación del paciente.

CareMe

En la aplicación del paciente es donde se encuentra la mayor parte del cómputo, ya que es sobre quien el sistema aprende.
Los procedimientos principales que se llevan a cabo en la aplicación son el de análisis y el de monitorización.

Análisis

El proceso de análisis solo se lleva a cabo una vez al día, cuando el teléfono entra en modo de carga. Para atender a esto, es necesario que el análisis lo realice una clase que extienda *BroadcastListener.* Este tipo de clases es necesario registrarlas en el manifiesto Android de la siguiente forma.

```
<receiver android:name=".PowerConnectionReceiver" >
   <intent-filter>
      <action android:name="android.intent.action.ACTION_POWER_CONNECTED" />
   </intent-filter>
</receiver>
```

En este caso, la clase que extiende se llama *PowerConnectionReceiver* y se registra para la llamada de conexión a la alimentación. El comportamiento a realizar debe escribirse en el método sobrescrito *onReceive(Context context, Intent intent),* que en la aplicación CareMe, realiza llamadas a distintos métodos para ir completando los pasos del análisis.

- Private void analyzeHistories(Context context, History h1, History h2)
  Usa las dos muestras de histórico que recibe como parámetros para crear una instancia de lugar si ha permanecido en el una estancia mínima, y en función de si se repite un número mínimo de veces, crear un lugar habitual. El número mínimo de repeticiones y la estancia mínima son variables globales que están definidas en tres repeticiones y quince minutos, este último expresado en milisegundos.

- Private int getInstancesCount(History h1)
  Es el método que usa *analyzeHistories* para realizar la cuenta de instancias de un lugar u en función de esta, crear un Lugar Habitual.

```
count = getInstancesCount(h1);
if (count > MIN_REPS) {
   //Comprobamos que no este ya en la bd e insertamos o actualizamos
   createPlace(context, h1, count);
}
```

- Private boolean createPlace(Context context, History h, int count)
  Es el método que usa *analyzeHistories* para crear un Lugar habitual. Primero busca si el lugar a crear ya existe, en cuyo caso realiza una media de las coordenadas para ser lo más preciso posible a medida que va a prendiendo e incrementa la asiduidad en uno.

```
if (l1.distanceTo(l2) < MIN_DISTANCE) {
    //hacemos media de la localización para ser cada vez mas exactos
    double avlat = ((l1.getLatitude() * p.getAssiduity()) + l2.getLatitude())
            / (p.getAssiduity() + 1);
    double avlon = ((l1.getLongitude() * p.getAssiduity()) + l2.getLongitude())
            / (p.getAssiduity() + 1);
    p.setLatitude(avlat);
    p.setLongitude(avlon);
    //actualizamos assiduity
    p.setAssiduity(p.getAssiduity() + 1);
    db.insertPlace(p);
    found = true;
}
```

Si no existe aún lo inserta en la base de datos. MIN_DISTANCE es la distancia que se usa para considerar si dos coordenadas son el mismo lugar, su valor se fija en sesenta metros.
El método devuelve *true* si el lugar ya existía.

- Private void analyzeInstances(Context context, PlaceInstance p1, PlaceInstance p2)
  Es el segundo método que invoca *onReceive* para los siguientes pasos del análisis. Tiene la misma estructura que *analyzeHistories*, pero esta vez con Instancias de Lugar como parámetro para construir Instancias de Rutinas y las propias Rutinas. También usa la estancia mínima para comprobar si entre los lugares hay un mínimo desplazamiento y disminuir los errores del GPS

- Private int getRInstancesCount(PlaceInstance p1, PlaceInstance p2)
  Tiene exactamente el mismo funcionamiento que *getInstancesCount.*

- Private void createRoutine(Context context, PlaceInstance p1, PlaceInstance p2, **int** count)
  Lo invoca *analyzeInstances* para crear Rutinas. Primero comprueba que la rutina no exista ya, pero esta comprobación tiene varias fases.

```
if (l1.distanceTo(l3) < MIN_DISTANCE && l2.distanceTo(l4) < MIN_DISTANCE) {
    routineFreqs = db.getRoutineFreqs(r.getId());
    for (int i = 0; i < routineFreqs.size() && !found; i++) {
        rf = routineFreqs.get(i);
        if (getWeekDay(p1.getStart()).equals(rf.getFrequency().getDay())) {
            //Actualizamos reliability
            if (rf.getReliability() < 95) {
                rf.setReliability(rf.getReliability() + 5);
                db.insertRoutineFreq(rf.getRoutine().getId(),
                        rf.getFrequency().getId(), rf.getReliability(), rf.getId());
            } found = true;
        }
    }
}
```

primero se comprueba que los lugares de partida y destino sean los mismos, tras esto, se deben hacer comprobaciones de la frecuencia. Si los lugares coinciden y la frecuencia también está ya incluida, se incrementa su fiabilidad. En caso de que no exista la frecuencia se añade, y en caso de que no exista la rutina, se crea con todas sus relaciones de lugares y frecuencia.

Monitorización

Este proceso debe estar en ejecución constante todo el día, independientemente de si la aplicación no esta abierta. Es por esto que se decidió implementarlos en un Servicio de Android. Esta clase debe extender de *Service* e implementar sus métodos de creación. También ha de ser registrado en el manifiesto como sigue.

```
<service
   android:name=".AccumulatorService"
   android:enabled="true"
   android:exported="true" >
</service>
```

Los métodos de creación, *onCreate()* y *onStartCommand(Intent, int, int)* se usan para crear la instancia de la base de datos y registrar los usuarios de la comunicación de Nimbees.

AccumulatorService, que es como se llama esta clase, realiza los procesos de acumulación de muestras de histórico y de monitorización.

La acumulación de histórico se lleva a cabo mediante un *LocationListener*.

```
locManager = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
Criteria crit = new Criteria();
crit.setAccuracy(Criteria.ACCURACY_FINE);
crit.setPowerRequirement(Criteria.POWER_LOW);
provider = locManager.getBestProvider(crit, true);
locManager.requestLocationUpdates(provider, 0, MIN_DISTANCE, locListener);
```

Se usan criterios para conseguir un compromiso entre un consumo de batería mínimo y precisión máxima. Recogemos muestras cada vez que se realiza un desplazamiento de MIN_DISTANCE, que eta definido globalmente a treinta metros.

Aprovechando estas ejecuciones, es cuando se envía la posición al cuidador y se lanza el proceso de monitorización.

- Public void sendPersonalizado(Location loc)
  Es el método que se llama al recoger la muestra de histórico para enviar la posición actual al cuidador. Este envío se realiza mediante una *CustomNotification* de Nimbees. En este mensaje se incluye un fragmento JSON con las coordenadas y se envía.

```
usuarios.add(email);
filters.add(new UserNameListFilter(usuarios));

//Envio un mensaje con los usuarios del filtro como content (Utilizo GSON para convertir
// List<String> a un String en formato JSON, y luego convertire ese String en List<String> con
//GSON)
NimbeesNotificationManager nimbeesNotificationManager =
      NimbeesClient.getNotificationManager();
Gson gson = new Gson();
nimbeesNotificationManager.sendNotification(gson.toJson(coordenadas),
      MessageContent.NotificationType.CUSTOM, filters, new NimbeesCallback())
```

- Private void monitor(Location loc)
  Esta función también se llama al recoger la muestra de histórico. Funciona solo si han pasado cinco minutos desde su última intervención. Cuando esto sucede comprueba si un contador que se incrementa cada vez que se guarda una muestra si al menos su valor es de seis, lo que supondría que en esos cinco minutos, el usuario se ha desplazado aproximadamente doscientos metros. En caso de que si, se actualiza la lista de rutinas que podría estar siguiendo el usuario (que son todas las que parten del mismo lugar que partió el usuario), y en caso de que no, querría decir que el usuario se paró y se resetea la lista.

```
if (act.getTime() - date.getTime() > 300000) {
   int medidas = shared.getInt("medidas", 0);
   if (medidas >= 6) {
      //Esta andando, Actualizo lista
      actualizarLista(loc);
   } else {
      //Se ha parado, Reseteo lista
      reseteaLista(loc);
   }
```

- Private void reseteaLista(Location loc)
  Regenera la lista de posibles rutinas que el usuario podría seguir saliendo de la localización loc. También se busca la que debería estar haciendo a esa hora ese día, si es que existe para tenerla en especial consideración.

- Private void actualizarLista(Location loc)
  Es el método que llama el monitor para ir actualizando la lista. La actualización de la lista consiste en ir eliminando de ella aquellas que ya no está siguiendo el usuario según el algoritmo. Es importante a la hora de ir

realizando las comprobaciones con las frecuencias y tiempos, realizar un traslado de la fecha en que se produjo la rutina y se almacenó a la fecha actual para poder extraer las horas de salida y llegada que hay que tener en cuenta en el algoritmo.

```
double v1 = (l1.distanceTo(l2) + 500) / (rlist.get(0).getRoutine().getEnd().getTime()
      - rlist.get(0).getRoutine().getStart().getTime());
long t1 = new Date().getTime() % 86400000;
long t2 = rlist.get(0).getRoutine().getEnd().getTime() % 86400000;
double v2 = (loc.distanceTo(l2)) / (t2 - t1);
if (v2 > v1) {
   ids.remove(id);
   if (id.compareTo(shared.getString("probable", Integer.toString(-1))) == 0) {
      shownot("Se ha salido de la rutina más probable!");
   }
}
```

Siendo en este fragmento l1 y l2 los lugares de salida y llegada respectivamente y loc la localización actual del usuario.
En los casos de riesgo, es decir cuando no sigue la rutina más probable, o ya no sigue ninguna, se lanza una notificación al usuario y se envía un mensaje al cuidador para alertarlo.

Anexo III

Publicación

Este trabajo ha servido de objeto para un articulo publicado en las jornadas SISTEDES de Ciencia e Ingeniería de Servicios JCIS 2015.
El articulo se basa en una descripción de este trabajo en las fases iniciales del mismo para ir dando a conocer el modelo de PeaaS sobre el que se ha trabajado en el presente proyecto.
A continuación se adjunta el articulo completo publicado.

SafeWalks: aplicación móvil de supervisión de pacientes de Alzheimer

Pablo Pérez Lozano[1], Alejandro Pérez Vereda[2],
Juan Manuel Murillo[1], Carlos Canal[2]

[1] Universidad de Extremadura; [2] Universidad de Málaga

pperezlo@alumnos.unex.es; apvereda@uma.es;
juanmamu@unex.es; canal@lcc.uma.es

Abstract. *El principal objetivo de Internet of Things (IoT) es integrar las tecnologías informáticas en el quehacer cotidiano de las personas, facilitando su interacción con un entorno de dispositivos interconectados, pero el estado actual del arte hace que dicha interacción esté aún lejos de resultar trivial, precisando de continua intervención del usuario. El modelo People as a Service (PeaaS) pretende facilitar estas tareas por medio del uso del teléfono móvil como interfaz del usuario con IoT. PeaaS permite elaborar un perfil sociológico del usuario, que puede ser explotado por el mismo y servido a terceros de forma controlada. En este trabajo presentamos una aplicación móvil para la supervisión de personas afectadas de alzheimer como prueba de concepto del modelo PeaaS, teniendo como resultado una funcionalidad que va mucho más allá de la ofrecida por otros productos similares en este campo.*

1 Introducción

El principal objetivo de *Internet of Things* (IoT) es integrar las tecnologías informáticas en el quehacer cotidiano de las personas [1, 2], de forma que, por ejemplo, se encienda de forma automática la cafetera para tener el café preparado al levantarnos, incluso planificando esta tarea de acuerdo con nuestros hábitos diarios.

Sin embargo, la forma en que esta integración se lleva a cabo en la actualidad es notablemente susceptible de mejora, quedando aún un salto entre la red donde la información es tratada e intercambiada y la realidad de la vida física y su contexto [3]. En general, el usuario necesita configurar diversos parámetros del sistema en cuestión de forma manual, lo cual, lejos de hacer que la tecnología trabaje para las personas, las obliga a estar pendientes de introducir nuevas órdenes o modificar la planificación siempre que se produzca un cambio inesperado en sus hábitos.

En un escenario de IoT más adecuado, la tecnología debería tener en cuenta el contexto de las personas a las que debe servir, aprendiendo de dicho contexto y realizando acciones de forma proactiva de acuerdo con su situación y expectativas en cada momento. Así, cuando una persona pone el despertador más tarde de lo habitual, le gustaría que el café empezará a hacerse en el momento adecuado para tomárselo caliente al levantarse.

Esto es lo que plantea el modelo *People as a Service* (PeaaS) [4]. PeaaS se basa en el uso de *smartphones* como representantes e interfaces de los usuarios con IoT, debido a que este tipo de dispositivos son altamente personalizados, y acompañan a sus propietarios a lo largo de sus actividades diarias y, en particular, en la mayor parte de sus interacciones en Internet, acumulando una gran cantidad de información sobre ellos [5,6]. PeaaS es un modelo social que pone su énfasis en el usuario como proveedor de servicios y le permite controlar la información que ofrece su dispositivo móvil, dando especial importancia a aspectos de privacidad y de seguridad.

Con la intención de validar este modelo, y como prueba de concepto del mismo, este trabajo se centra en el desarrollo de una aplicación Android para el seguimiento de personas con capacidades intelectuales mermadas, pero con cierto grado de autonomía, como pueden ser las personas con alzheimer. El objetivo es que el teléfono móvil aprenda los patrones habituales del usuario (horarios, rutinas de desplazamiento, etc.), de forma que pueda supervisar dichas actividades y realizar acciones (por ejemplo, alertas al paciente o a sus familiares) en caso de un cambio inesperado en dichos patrones. La propuesta entronca con el interés actual en el desarrollo de productos software relacionados con la salud de los usuarios. Es tal el crecimiento de este sector que incluso Google ofrece una API[1] para la creación de aplicaciones destinadas a fomentar hábitos saludables de una forma más fácil, rápida y eficiente.

La estructura de este artículo es la siguiente. La Sección 2 presenta y discute la motivación del trabajo y su campo de aplicación. La Sección 3 desarrolla los principales aspectos técnicos del mismo, incluyendo entre otros la descripción del caso de estudio, la arquitectura de la aplicación, el procedimiento de análisis de detección de rutinas y el estado actual del producto. La Sección 4 hace un repaso de los principales trabajos relacionados. Por último, la Sección 5 recoge las conclusiones de este trabajo.

2 Motivación

Este trabajo se enmarca en el desarrollo de una plataforma software que mejore cómo las personas se integran en IoT, para lo que se hace uso de la información contextual y perfil de usuario alojado en sus teléfonos móviles, paliando así las limitaciones actuales de este tipo de sistemas. Esto abre camino a la transición del actual modelo de IoT hacia el que podríamos llamar *Internet of People* (IoP) [7].

Esta línea de investigación parte de trabajos anteriores sobre los modelos y plataformas Social Devices [8,9] y PeaaS. Social Devices es una plataforma desarrollada en la Universidad de Tampere que tiene por objeto enriquecer, incrementar y facilitar las interacciones entre personas geográficamente próximas y entre ellas e IoT. Por otro lado, PeaaS, es un modelo y plataforma de computación móvil que permite la gestión de perfiles sociológicos de los usuarios, que son inferidos y almacenados en el propio dispositivo móvil y proporcionados a terceros como un servicio en la Nube, de manera segura y controlada por su propietario. La combinación de ambos modelos permite convertir el teléfono móvil en la interfaz

[1] Google Fit. http://developers.google.com/fit/

natural entre las personas e IoT, de forma que este almacena toda la información contextual necesaria e interactúa con IoT en consecuencia, minimizando la necesidad de intervención del usuario.

En la actualidad, con la aparición en el mercado de diversos dispositivos *wearable*, el sector de IoT está experimentando un gran crecimiento y difuminando las fronteras que había en cuanto a lo que es posible hacer con estas tecnologías. La mayoría de estos dispositivos salen al mercado en una línea claramente dirigida hacia sanidad y hábitos saludables. Esto es lo que nos ha llevado a elegir este sector como la prueba de concepto más adecuada.

Existen en la actualidad diversas aplicaciones móviles para la ayuda de personas con alzheimer, entre las que podemos citar Cerqana[2] y Tweri[3]. Sin embargo, ninguna de ellas considera el aprendizaje automático de rutinas y patrones de desplazamiento. Esto sin duda es una gran ventaja de nuestra aplicación sobre las ya existentes, ya que minimiza la necesidad de configuración de distintos parámetros que pueden ser complejos. De esta manera se facilita el uso de la aplicación tanto a los enfermos, que suelen tener edad avanzada y no tienen interiorizado el uso del teléfono, como a sus cuidadores, que simplemente recibirán notificaciones cuando sea oportuno. Para ello, nuestra propuesta sigue el modelo de la pirámide DIKW, que distingue entre los niveles de datos, información, conocimiento y sabiduría [10]. Así, partiendo de la recolección y agregación de datos de geolocalización temporal del teléfono móvil, se obtiene información sobre lugares y trayectorias habituales del usuario. A partir de esta información, es posible inferir conocimiento sobre sus rutinas de desplazamiento y frecuencia de las mismas, adquiriendo finalmente la sabiduría necesaria para poder predecir comportamientos y detectar desviaciones en los mismos, tomando decisiones de emisión de alertas en caso necesario.

3 Una aplicación para enfermos de alzheimer

El escenario elegido como prueba de concepto del modelo PeaaS consiste en el desarrollo de SafeWalks, una aplicación para ayudar a personas con alzheimer en las fases iníciales de esta enfermedad. Con ella se quiere contribuir a mejorar su calidad de vida y la de sus cuidadores. El síndrome de Alzheimer, es una enfermedad degenerativa que produce un deterioro de las neuronas del cerebro. Esto afecta al enfermo con síntomas como pérdida de memoria, desorientación o pérdida de otras capacidades mentales que pueden resultar muy peligrosas para su seguridad. Estos síntomas pueden manifestarse con mayor fuerza en ciertos momentos, afectando a la persona de inmediato. Por ejemplo, el enfermo ha salido de casa a realizar alguna actividad y durante ella olvida el propósito de lo que estaba haciendo o cómo volver a casa.

Existen alrededor de 24 millones de personas afectadas por alzheimer en el mundo, y se estima que esta cifra alcanzará los 80 millones dentro de 20 años. Esta enfermedad no solo afecta a los pacientes, sino que también condiciona la vida de sus cuidadores, ya que les obliga a estar constantemente pendientes de ellos y

[2] Cerqana. http://www.cerqana.es
[3] Tweri. http://www.tweri.com

preocupados por su situación. Con objeto de contribuir a la mejora de esta situación, nos planteamos el desarrollo de una aplicación para dispositivos móviles basada en PeaaS y capaz de aprender las rutinas de desplazamiento del usuario**.** La motivación para ello fue doble. Por un lado la penetración cada vez mayor de los dispositivos móviles incluso entre la población mayor. Por otro, las ventajas que proporciona PeaaS para potenciar el uso del contexto de los usuarios de dispositivos móviles.

La aplicación diseñada incorpora dos componentes funcionales:

***Dispositivo Paciente*:** Es el componente principal y se trata de una aplicación Android que reside en el dispositivo del paciente. Valiéndose de la información facilitada por el GPS del dispositivo, la aplicación registra los desplazamientos del usuario y los analiza para aprender y detectar sus rutinas. Comprueba continuamente si los desplazamientos realizados son acordes a dichas rutinas, sino emite avisos tanto al paciente como a su cuidador según diferentes niveles de alarma que pueden ser configurados. SafeWalks utiliza, analiza y pone al servicio de otros su contexto. La aplicación también permite indicar qué cuidadores tendrán acceso a este contexto.

Dispositivo Cuidador: De nuevo es una aplicación Android, que en este caso reside en el dispositivo del cuidador. A través de ella el cuidador puede realizar un seguimiento de la posición del paciente y configurar de forma remota la aplicación de este indicando los niveles de alarma deseados y las actuaciones a llevar a cabo en la detección de cada nivel. La aplicación se encarga de recibir y procesar las alarmas dándoles el tratamiento adecuado.

3.1 Arquitectura

En esta sección se describe la arquitectura adoptada para integrar PeaaS en el desarrollo de la aplicación. La Figura 2 muestra dicha arquitectura.

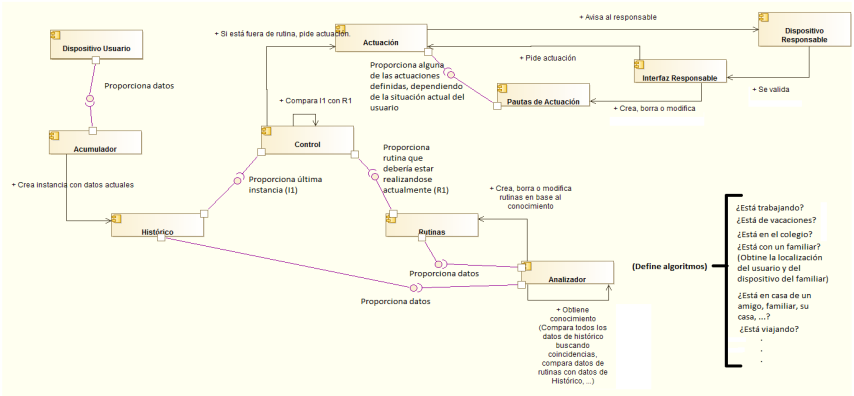

Fig. 1. Arquitectura de componentes

El componente *Acumulador* se encarga de registrar los datos de desplazamiento del paciente. Para ello detecta cuándo el paciente se pone en movimiento y, a partir de ese momento y hasta que de nuevo entre en reposo, va registrando sus coordenadas GPS, velocidad y dirección, con una cadencia variable. Toda esta información es

facilitada al *Histórico* que se encarga de almacenarla. No obstante, el componente principal de la arquitectura es el *Analizador* que se encarga de analizar los datos de desplazamiento almacenados por el *Histórico* para detectar las rutinas del paciente. El algoritmo seguido para ello se detalla en la Sección 3.3. Las rutinas son almacenadas como hitos en una línea de tiempo que coincide con un día de actividad.

Una vez establecidas las rutinas del paciente, el componente *Controlador* se encarga de analizar si su actividad actual corresponde a una rutina registrada o no**.** Para ello, simplemente toma la siguiente rutina en la línea de tiempo y, una vez alcanzado el momento del tiempo indicado por la rutina analiza la última información almacenada en *Histórico* y comprueba si son similares según unos márgenes de tiempo y espacio configurables. En caso de detección de una actividad fuera de rutina se notifica el hecho al componente *Actuador*. Este componente tiene configurados diferentes *Tipos* de desviación de rutinas. Por ejemplo, el cuidador puede establecer que encontrarse lejos del lugar habitual a las 12:00 del mediodía es una desviación de rutina de índole distinta a si el mismo hecho se produce a las 2:00 de la madrugada. Para cada *Tipo* de violación se pueden configurar diferentes *Niveles* de alarma. Cada nivel puede llevar asociadas acciones que van desde el lanzamiento de eventos en el propio dispositivo móvil del paciente hasta el lanzamiento de llamadas telefónicas automáticas al cuidador o incluso el lanzamiento de mensajes sonoros desde el móvil del paciente para captar la atención de las personas que puedan encontrarse cerca. Toda esta información es gestionada por el componente *Pautas de Actuación*. El cuidador puede tomar control de las alarmas en el momento que lo desee gracias a los servicios ofrecidos por el componente *Interfaz Cuidador*.

3.2 Procedimiento de análisis

Como mencionamos anteriormente, el componente principal de la arquitectura es el *Analizador*, que revisa los desplazamientos que realiza el paciente y trata de descubrir y aprender cuáles son sus rutinas. Para ello, este componente lanza su análisis una vez al día a una hora preestablecida. Inicialmente, para el diseño y construcción de este componente se valoró la posibilidad de utilizar motores de inferencia existentes disponibles incluso para dispositivos móviles, en particular AndroJena[4]. Sin embargo, tras valorar su potencial se optó diseñar y construir un procedimiento de análisis y aprendizaje *ad hoc*.

El primer paso en el análisis es aislar en el *Histórico* los desplazamientos correspondientes al último día tal y como esquematiza la Figura 2.a. Para ello, se recorre el histórico hacia atrás, tratando de localizar el primer desplazamiento de la mañana (M1 - primer desplazamiento tras un período de inactividad coincidente con la fase nocturna). A partir de este hito se sigue recorriendo el *Histórico* para localizar el siguiente primer movimiento de la mañana (M2). La actividad del último día queda descrita por los desplazamientos registrados entre M2 y M1.

[4] AndroJena: http://code.google.com/p/androjena/

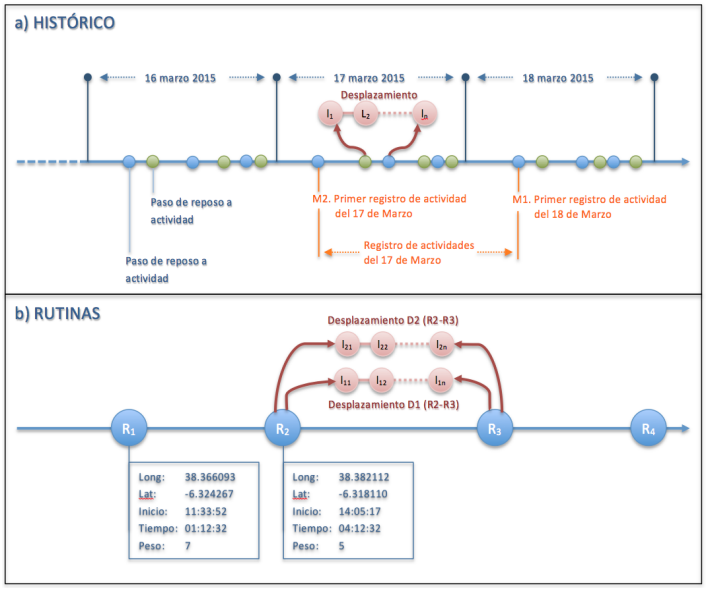

Fig. 2.a Almacenaje y procesamiento de *Histórico* y *Rutinas*

El siguiente paso es llevar la información contenida entre M2 y M1 a *Rutinas*, la línea de tiempo que recoge la actividad del paciente durante una jornada, y que sirve para chequear si el paciente está realizando sus rutinas o se encuentra fuera de ellas. El algoritmo para ello se basa en el siguiente pseudocódigo:

```
1     Recorrer HISTÓRICO desde M2 hacia M1
2          Para cada periodo P de inactividad detectado
3               RUT <- Recorrer RUTINAS buscando periodo de
                       inactividad similar a P atendiendo en cuenta
                       los márgenes de espacio y tiempo configurados
4               Si RUT <> Null entonces
5                    RUTINAS.RUT.PESO <- RUTINAS.RUT.PESO + 1
6               Sino
7                    RUTINAS.NUEVA_RUTINA (P)
```

Fig. 2.b Almacenaje y procesamiento de *Histórico* y *Rutinas*

El algoritmo localiza en primer lugar los periodos P de inactividad del paciente y, para cada uno de ellos, se determina sus coordenadas y la duración de la inactividad. A continuación, para cada P encontrado se recorren las *Rutinas* del paciente tratando de encontrar un periodo de inactividad similar. En *Rutinas* cada periodo de inactividad se recoge como un hito registrando su longitud, latitud, hora de comienzo, duración y peso tal y como ilustra la Figura 2.b. El peso indica el número de días que se ha repetido dicha rutina. Para realizar las comparaciones se utilizan unos márgenes de tolerancia de espacio y tiempo configurables. Si se encuentra un hito similar en la línea de tiempo se suma una unidad a su peso. En caso contrario se ha encontrado una nueva rutina y se añade un nuevo hito a la línea de tiempo con peso 1. Con ello, y a medida que la aplicación lleva más días de funcionamiento, se consigue reforzar los

hitos que corresponden a rutinas más asiduas con mayores pesos. Esto también permite establecer distintos niveles de alarma dependiendo del peso de la rutina.

Las rutas de desplazamiento entre puntos de inactividad reciben un tratamiento similar. Tal y como ilustra la Figura 2.a, cuando el paciente está en movimiento el dispositivo móvil registra en el *Histórico* su desplazamiento almacenando la latitud, longitud, dirección y velocidad con una cadencia que de nuevo es configurable en la aplicación. Cadencias de un registro cada 3-5 minutos son aceptables. *Rutinas* no solo almacena los puntos de inactividad sino también todas las rutas que el *Paciente* recorre entre ellos. Entre dos rutinas puede haber más de una rutas, como muestran D1 y D2 en la figura 2.b. Para cada Desplazamiento registrado en *Histórico* durante el último día se trata de localizar su existencia en *Rutinas*. La búsqueda ahora es mucho más dirigida dado que se conocen las rutinas R de origen y destino. Sólo hay que comparar la nueva ruta entre el origen y destino con cada una de las rutas almacenadas en Rutinas para ese origen y destino. De nuevo la comparación se realiza atendiendo a unos márgenes de tolerancia de espacio configurables. Más concretamente la comparación se realiza analizando si los puntos de la de la nueva ruta (l_1, l_2,...l_n) se encuentran en la ruta almacenada en *Rutinas* (por ejemplo l_{11}, l_{12},...,l_{1n}) o en sus puntos intermedios (por ejemplo entre l_{11} y l_{12}).

3.3 Optimización del consumo de recursos

En el diseño de aplicaciones destinada a dispositivos móviles, la optimización del consumo de recursos del dispositivo debe ser un factor director en su diseño.

En primer lugar, teniendo en cuenta el tamaño limitado de la memoria de los dispositivos, se ha optado por no guardar el historial de desplazamientos del paciente por un plazo indefinido. Como ya se ha mencionado anteriormente, una vez que los datos correspondientes al último día de actividad han sido procesados e integrados a la línea de tiempo del paciente podrían eliminarse. Aún así se permite la opción de almacenar un número de días de historia. Esto se hace con el objetivo de permitir regenerar la línea de tiempo del paciente a partir de esos días de historia. Esto resulta beneficioso cuando el paciente introduce un cambio en sus rutinas de comportamiento y desea regenerar su línea de tiempo a partir de las actividades realizadas en los últimos días. Por ejemplo, imagínese que el paciente cambia su domicilio habitual por una residencia con el objeto de conseguir asistencia durante todo el día.

El procesamiento de la actividad de la jornada, el reconocimiento de rutinas y su integración en la línea de tiempo es una tarea que consume muchos ciclos de reloj y en consecuencia batería. Por ello esta actividad se lleva a cabo sólo cuando se detecta que el teléfono está en carga conectado a la red eléctrica.

Por último, todas las actividades de registro de localización física también consumen batería y memoria. Sin embargo en este caso resulta difícilmente evitable. Si se desea mucha precisión en los registros de información y en los cálculos de cercanía se requieren cadencias de registro muy elevadas. No obstante, como ha podido comprobarse, existen gran número de parámetros configurables en la aplicación con el objeto de poder adecuar estas cadencias a las características de cada paciente.

Actualmente se dispone de un prototipo funcional de la aplicación. Dicho prototipo monitoriza la actividad del paciente, aprende sus rutinas en base a puntos de inactividad y rutas entre ellos y realiza detecciones básicas de si el paciente se encuentra en rutina o no. Sin embargo, todavía no se ha implementado todo el tratamiento para las alertas, que se encuentra actualmente en construcción. Tras completar dicha construcción se procederá a una evaluación sistemática de la aplicación y a estudiar cuáles son los parámetros de configuración más adecuados en función del tipo de paciente con el objeto de optimizar las detecciones de alertas eliminando falsos positivos y negativos así como el consumo de recursos.

4 Trabajos relacionados

PeaaS es un modelo que persigue poner en valor el contexto de las personas proporcionándolo como un servicio desde el teléfono móvil. Tecnologías como Siri[5] para iOS o Sherpa[6] para Android contribuyen en el mismo sentido proporcionando a los dispositivos móviles capacidades atribuibles a las personas. PeaaS contribuye a iniciar el siguiente paso en el que dichas capacidades se adaptarán a la personalidad y preferencias de cada usuario.

La aplicación de PeaaS en este trabajo consiste en poner el contexto del *Paciente* al servicio del *Cuidador*. Ya existen otras aplicaciones móviles basadas en una arquitectura convencional que asisten a pacientes de alzheimer con el mismo objetivo que SafeWalks. Una de las pioneras en este género es la ya mencionada Tweri. Esta aplicación proporciona funcionalidades básicas para identificar las zonas donde el paciente está seguro de forma que se lancen alertas al cuidador si las abandona. El cuidador puede seguir al paciente a través de una interfaz. Además, si el paciente se encuentra inseguro pude lanzar una alerta al cuidador mediante una pulsación en la interfaz. Con funciones mejoradas y el mismo objetivo, Cerqana permite la fijación manual de zonas seguras y peligrosas en un mapa, aviso vía notificación si entra en alguna zona marcada como insegura, fijación manual de rutas habituales del usuario o detección de caídas o desvanecimientos. En comparación con SafeWalks, ambas aplicaciones adoptan una estrategia en la que el dispositivo móvil del paciente se encarga de subir su localización a un servidor con una cadencia determinada. En el servidor se realizan todos los análisis de datos y el tratamiento de alertas. Ninguna de ellas aborda la detección y aprendizaje de las rutinas del paciente, ni la actuación en tiempo real ante alertas. Simplemente está basada en los datos que se encuentran en el servidor y que normalmente tienen minutos de retraso que, en algunos casos, pueden resultar decisivos. Esta característica es difícilmente solventable mediante el uso de arquitecturas convencionales ya que implicaría cadencias muy altas en la subida de datos al servidor con el consiguiente consumo de batería. Plataformas como UrbanAirship[7] advierten

[5] Siri. https://www.apple.com/es/ios/siri/
[6] Sherpa. http://sher.pa/
[7] UrbanAirship: http://urbanairship.com/

que cadencias de registro y subida de datos cercanas a los 30 segundos implican el consumo de la batería del dispositivo en un periodo de 30 a 40 minutos.

Finalmente, se ha optado por emplear un algoritmo propio y muy básico para el análisis y aprendizaje de rutinas. Sin duda, el uso de herramientas como RDFQuery[8], RDFStoreJS[9], Nools[10] o la ya mencionada AndroJena podrían contribuir a mejorar nuestra propuesta. Su uso está contemplado en un siguiente paso cuando se proceda a incorporar datos al análisis más allá de la localización.

5 Conclusiones

El progresivo desarrollo de IoT y la salida al mercado de nuevos dispositivos *wearable* están haciendo cambiar la forma en que interactuamos con nuestro entorno, favoreciendo el surgimiento de escenarios cada vez más interconectados y de complejidad creciente. Sin embargo, el estado actual de la tecnología, obliga a una continua intervención del usuario. Modelos como PeaaS abogan por el uso del teléfono móvil como interfaz del usuario con su entorno, y permiten el desarrollo de perfiles sociológicos que pueden ser ofrecidos como servicios a terceros de forma controlada, favoreciendo la transición de IoT a IoP.

En este trabajo presentamos el desarrollo de una aplicación para la supervisión y seguimiento de enfermos de alzheimer, como prueba de concepto del modelo PeaaS. A partir de la monitorización de la señal GPS del teléfono, la aplicación desarrollada es capaz de aprender las rutinas de desplazamiento del usuario y de tomar decisiones en caso de desviaciones de las mismas que puedan implicar un peligro para el enfermo. SafeWalks ofrece una funcionalidad significativamente superior a otras similares actualmente en el mercado, y muestra cómo los conceptos inherentes al modelo PeaaS pueden ser puestos en práctica en un escenario de uso real, más allá del caso de estudio elegido como prueba de concepto. Actualmente estamos culminando el desarrollo de la funcionalidad básica de la aplicación y estudiando las posibilidades de su comercialización.

En cuanto a trabajos futuros, cabe señalar la incorporación progresiva de nuevos algoritmos de aprendizaje, para lograr una mayor precisión tanto en la detección de rutinas de desplazamiento, como en la predicción de las mismas y en la detección de desviaciones respecto al comportamiento habitual. La integración con servicios como Google Maps permitiría obtener información de interés sobre el entorno de las coordenadas del usuario, afinar la detección de situaciones peligrosas y proporcionar ayuda al usuario para retomar su rutina de desplazamiento en caso de que se haya extraviado. Por otro lado, el uso de la señal *wifi*, y *bluetooth* del teléfono permitiría un mejor seguimiento de la actividad del usuario en interiores, por ejemplo mediante iBeacons[11], y la detección de si está acompañado por sus cuidadores o no.

[8] RDFQuery: https://code.google.com/p/rdfquery/
[9] RDFStoreJS: https://github.com/antoniogarrote/rdfstore-js
[10] Nools: https://github.com/C2FO/nools
[11] iBeacon. http://developer.apple.com/ibeacon/

Agradecimientos

Este trabajo ha sido parcialmente subvencionado por el Gobierno de España, mediante el proyecto TIN2012-35669, mediante la Red de Excelencia en Ciencia e Ingeneiría de Servicios TIN2014-53986-REDT, mediante la Red de Excelencia en Ingeniería del Software Dirigida por Modelos TIN2014-53555-REDT y por la Universidad de Málaga, Campus de Excelencia Internacional, Andalucía Tech.

Referencias

[1] M. Bakardjieva, R. Smith. "The Internet in everyday life computer networking from the standpoint of the domestic user": New Media & Society, 3(1):67-83, 2001.

[2] Dohr, R. Modre-Opsrian, M. Drobics, D. Hayn, G. Schreier. "The Internet of Things for ambient assisted living": Information Technology - New Generations (ITNG), Seventh International Conference on, pp. 804-809, IEEE Computer Society, 2010.

[3] L. Sha, S. Gopalakrishnan, X. Liu, Q. Wang. "Cyber-physical systems: A new frontier": Machine Learning in Cyber Trust, pp. 3-13, Springer, 2009.

[4] J. Guillén, J. Miranda, J. Berrocal, J.A. García-Alonso, J.M., Murillo, C. Canal. "People as a Service: A Mobile-centric Model for Providing Collective Sociological Profiles": Software, IEEE Software, 31(2):48-53, IEEE Computer Society, 2014.

[5] L Srivastava "Mobile phones and the evolution of social behavior": Behaviour & Information Technology, 24(2), 2005.

[6] Oulasvirta, T. Rattenbury, L. Ma, E. Raita "Habits make smartphone use more pervasive": Personal and Ubiquitous Computing, 16(1):105-114, 2012.

[7] J. Miranda, N. Mäkitalo, J. García-Alonso, J. Berrocal, T. Mikkonen, C. Canal, J.M. Murillo. "From the Internet of Things to the Internet of People", IEEE Internet Computing, 19(2):40-47, IEEE Computer Society, 2015.

[8] N. Mäkitalo, J.Pääkkö, M. Raatikanien, V. Myllärniemi, T. Aaltonen, T. Leppänen, T. Mänistö, T. Mikkonen, "Social Devices: Collaborative Co-located Interactions in a Mobile Cloud": Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia, MUM, 2012.

[9] N. Mäkitalo. "Building and programming ubiquitous social devices": Proceedings of the 12th ACM international symposium on mobility management and wireless access (MobiWac'14), pp. 99-108, ACM 2014.

[10] T.H. Davenport, L. Prusak. *Working Knowledge : How Organizations Manage What They Know*. Harvard Business School Press, 1998.